An Empirical Assessment of the Software Design Pattern Concept

CHENG ZHANG

An Empirical Assessment of the Software Design Pattern Concept

Cheng Zhang

School of Engineering and Computing Sciences

Durham University

Ph.D. Thesis

2011

I would like to dedicate this thesis to my loving parents,
Deping Zhang and Shiping Wan,
who offered me unconditional love and support
throughout the course of this thesis.

Acknowledgements

Foremost, I would like to express my deep and sincere gratitude to my supervisor, Professor David Budgen, for his guidance, wisdom, support, patience and immense knowledge during the drafting and research for this thesis.

I would like to thank Professor Barbara Kitchenham for her help during my study, and for acting as an external assessor for the survey.

I would like to thank Professor Ray Welland for acting as an external assessor for the survey, and also thank for his time and effort in examining this thesis.

I would like to thank Dr. Andrew Hatch for his time and effort in examining this thesis.

I would like to thank all my friends, but especially Florence, Daniel, and Alex, who supported and encouraged me during the long course of this thesis.

I would also like to thank all those who reviewed our work and participated in our survey, especially for those who provided us with their comments.

Last but not the least, I would like to thank my father, Deping Zhang, and mother, Shiping Wan, for giving birth to me and to both of them for supporting me throughout my life.

Abstract

Context: The publication of the milestone textbook on design patterns by the 'Gang of Four' (*GoF*) in 1995, introduced a set of 23 design patterns that are largely concerned with improving the practices and products of software development. However, there has been no comprehensive assessment of the effectiveness of design patterns, nor is there any evidence about any claims and factors that are made for pattern reuse in software development.

Aims: The aims of this thesis are to assess the design patterns systematically in a sequence of studies, and to identify the claims and factors to determine how well they reflect experiences of pattern reuse in practice.

Method: This thesis describes four studies: a document survey to identify claims for patterns, a mapping study to identify empirical studies about patterns, an online survey, and a narrative synthesis. The mapping study and the online survey together provide quite comprehensive and thorough evidence for the narrative synthesis. In the narrative synthesis, we check whether there is any consistency or not in the evidence about specific patterns, and also to see how the claims and factors influence pattern reuse.

Results: The mapping study found 20 primary studies, and the online survey had 206 usable responses. In the 20 primary study of the mapping study 17 design patterns were examined. In the online survey 175 respondents considered patterns were useful, and 155 respondents reported on patterns that they considered not to be useful.

Conclusion: From the synthesis results, the specific patterns Composite and Observer are evaluated as being generally useful, but the Visitor and Singleton patterns, while useful, have possible negative aspects. And also four of the claims and the effect of one factor are demonstrated to be generally true. But the others are either unsupported or have no effect.

Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

The objectives of this chapter are to introduce the motivation of this thesis and explain how the research questions were generated. Then it describes the contribution of this thesis, the success criteria, and also the overall structure of this thesis.

1.1 Software Reuse and Design Patterns

Software has become an indispensable product in people's work and daily life. It covers every field from agriculture to economy, from clinical to education. It provides particular services to satisfy specific customer needs and areas (Sommerville, 2007). With software performing such a key role, it has also become an essential product around the world. But not all software artefacts can be regarded as high quality products. High quality software should satisfy a variety of properties. Pree (1995) has suggested that *reusability* is one quality dimension of software. This property of reusability requires that a software system or its parts can be reused for other software development.

Because software reuse can potentially reduce the risk and cost for the software development process, reuse has become a distinct theme within software engineering (Mili Hafedh, 2002). Nowadays there are many established techniques for encouraging software reuse. Sommerville (2007) lists eleven techniques, such as design patterns, component-based development, application frameworks, etc.

Among these the use of design patterns is regarded as being mainly focused on object-oriented software design.

The concept of patterns that has become familiar to software developers is not from the area of software engineering. The contemporary concept of 'patterns' as a means of structuring design ideas was introduced by the architect Christopher Alexander in the 1970s (Alexander *et al.*, 1977). Alexander created a series of patterns for building design. With the popularity of this concept, the concept of a pattern has been employed in a number of domains such as computer science and art (Buschmann *et al.*, 1996). With respect to the domain of computer science, and software engineering in particular, the design pattern concept can be applied in the areas of software design, configuration management, user interface design and interaction scenarios (Sommerville, 2007). For software engineering, interest in the use of design patterns began in the 1980's when a set of patterns for developing elegant user interfaces in Smalltalk was developed by Ward Cunningham and Kent Beck (Beck, 1988), after which, Jim Coplien developed a set of patterns in C++ (Coplien, 1991). From the 1990s, starting with the well-known textbook by the 'Gang of Four' (Gamma *et al.*, 1995), often abbreviated to *GoF*, the literature on design patterns and their use in software design has become quite extensive.

A design pattern describes a recurring problem and its successful solutions with a high level of abstraction and forms a fundamental means of design reuse in OO development (Sommerville, 2007). A pattern names, abstracts, and identifies the reusable solution, based on experts' experiences. A regular template is used to describe when it can be applied, whether it can be applied in view of other design constraints, and any consequences and trade-offs involved in its use. The template for design patterns used in (Gamma *et al.*, 1995) covers the headings:

- *Pattern Name and Classification*,
- *Intent*,
- *Also Known As*,

- *Motivation*,
- *Applicability*,
- *Structure*,
- *Participants*,
- *Collaborations*,
- *Consequences*,
- *Implementation*,
- *Sample Code*,
- *Known Uses*,
- *Related Patterns*.

Design patterns provide access to design knowledge through the description of a set of successful solutions extracted from previous experiences with respect to a certain type of problem. But an important difference between using a design pattern and personal reuse from experience is that a design pattern provides a systematic catalogue (Beck *et al.*, 1996), so extending the scope of reuse beyond one person's experience and allowing the user of the pattern to draw upon the experiences of others.

1.2 Motivation for Thesis

The concept of design patterns has been widely employed in system creation after the publication of the textbook *GoF* (Gamma *et al.*, 1995). With the process of learning and using patterns, we became interested in why people use patterns and how they use them. These motivated the different forms of further investigation described in this thesis.

- Why people use design patterns.

From the concept of design patterns, it has two motivating elements in the definitions: 'design patterns are successful solutions' and 'design patterns solve recurring problems'. In theory these elements answer the question of 'why people reuse patterns', and encourage people to reuse them.

For the first element 'design patterns are successful solutions', developers undoubtedly expect to improve their products successfully via reusing successful solutions of design patterns. The textbook by the 'Gang of Four' (Gamma *et al.*, 1995), also suggested that the use of design patterns could help developers to improve their software products in terms of software quality. However this ideal expectation is based on effective use of design patterns.

As described above, a design pattern is one of the techniques for reuse in Sommerville (2007)'s list. Reusing design patterns also has the same problem with other techniques. Jones (2009) has indicated that successful reuse is determined by the quality of the reusable material. A poor quality reusable material can lead to a very negative effect upon system design. Jones (2009) also reported that currently the average volume of high-quality reusable material is less than 25%. As a reuse technique, design patterns also fit this condition. We cannot be sure how effective design patterns are, before a systematic investigation is conducted. Therefore a systematic investigation is necessary for us to assess the effectiveness of design patterns.

For the second element 'design patterns solve recurring problems', design patterns are means of sharing and transferring experiences of software design between designers and learners. But Sommerville (2007) is convinced that design patterns are only suitable for use by experienced developers. This made us doubt whether reusing design patterns can be used to solve recurring problems under any condition. Therefore it has motivated us to investigate people's experiences to see how far Sommerville is right. And also we are interested to identify any other factor that could influence the effectiveness of pattern reuse.

- How people use design patterns.

Although software engineering is intrinsically an 'empirical' subject, this aspect has so far largely been demonstrated through the (relatively informal) reuse of 'experience' rather than through the use of the outcomes from systematic empirical investigations. Various authors have noted this: Whitley (1997) commented that "a common pattern in software engineering research is the development of system-building techniques, such as object-oriented design, which are strongly advocated in the absence of evidence". Stemming from their systematic surveys of the literature. Glass and co-workers have made similar observations about the emphasis upon advocacy and the predominance of analysis as the means for evaluation (Glass *et al.*, 2002, 2004).

We have observed that the use of software design patterns for designing object-oriented (OO) systems would appear to fit into this category. Leaving our previous doubt, the concept of the design pattern assumes that knowledge of software design is transferable, it can be reused between different software designers. We expected to find guidance about how to use design patterns efficiently. However, from our primary study about design patterns it appears that the available literature on patterns is largely in the form of 'advocacy' or 'experience', and also is focused upon identifying and documenting patterns rather than providing guidance about *using* them. While the textbook by the 'Gang of Four' (Gamma *et al.*, 1995) employed a template for cataloguing patterns that has been widely adopted, the template offers little that would encourage the notion of providing evidence about where and when patterns might best be used, beyond the headings 'Applicability' and 'Known Uses'.

1.3 Statement of the Problem

The first and most basic problem is that design patterns have not been investigated, evaluated and validated systematically. Sometimes software reuse is regarded as a panacea (Jones, 2009). As a reuse technique the concept of 'patterns' has become popular in a number of domains (Buschmann *et al.*, 1996) and has been widely advocated for designing software systems. If designers employ patterns just to facilitate reuse, this can lead to 'blind' use and course a possible

poor product.

Indeed, in some studies about real development, the results indicate that design patterns do not always lead to positive products (Vokáč *et al.*, 2004; Wendorff, 2001). For some specific patterns, for example, the Singleton pattern, even one of the authors of the GoF has doubts about the benefit of it [1]. Perhaps a negative experience of using one pattern can make designers totally ignore patterns without being familiar with them.

These conflicting conditions reveal that there is no thorough investigation on design patterns. Most people have no comprehensive cognition for all design patterns. Therefore, a systematic investigation, assessment and validation for all design patterns is required.

The second problem is that we do not have enough evidence about what claims and factors about using design patterns influence software design. Moreover we have no evidence about how these potential claims and factors influence software design. From our primary study researchers and developers have become interested in using and writing design patterns for different aspects of software engineering. When doing so, it is intended that these design patterns will be shared with other developers for reuse in later development. But where do good design patterns come from? Schmidt *et al.* (1996) indicates that the good design patterns are claimed to originate from the practical experiences of software developers. However, there is no evidence to reveal what claims and factors decide a good pattern.

1.4 Thesis Structure

Figure 1.1 displays the interconnection between the chapters in this thesis. It is not only a process description, more importantly it displays the connections between the methods which are employed in the thesis. It shows that how we

[1] http://www.informit.com/articles/article.aspx?p=1404056

solve the problems and achieve our contributions via these methods. In this figure the oblong boxes represent the initial evidence and the final evidence we have obtained. The rectangle boxes represent the methods we have employed. The diamond boxes represent the outcomes those are generated by using those methods.

Figure 1.1: Thesis Structure

As shown in this figure, the literature about design patterns holds much of the evidence we have used. It provides the basis for our two methods which are based on the literature, including the literature survey about design patterns and the mapping study. We began with an informal survey that was employed it to investigate the claims about using patterns from some textbooks and the papers collected from the mapping study. The mapping study, as another method which is based on literature, is different with that literature survey. It is a formal and systematic scoping review (Kitchenham *et al.*, 2010). Here we employed it to help identify those primary studies that evaluate aspects of design patterns in any way and hence to determine what forms and issues have been studied, as well as by what means.

Both the literature survey and the mapping study identified some claims about using design patterns. These claims were regarded as the rationale for using design patterns in theory. But actually they were not validated. As a part of our thesis, they were synthesised with other studies in the next stage for validation.

The mapping study provided further empirical evidence about people's practical reuse of pattern. And also we summarised people's opinions on reusing patterns from the empirical papers, and identified which evidence was lacking in the outcomes of the review.

To obtain the practical assessments for the specific patterns based on people's experiences, the online survey was created based on the empirical evidences and the claims. In Chapter 7 we obtained the developers' views on the specific patterns. And also we extracted a factor for using patterns. This is similar with the claims. We have validated it in the stage of evidence synthesis (Chapter 8).

The evidence synthesis is described in Chapter 8. From the synthesis we can evaluate some specific patterns to see their effectiveness. And also the claims were checked and one relevant factor was identified. We can have the evidence about which claims and factor and how much they influence pattern reuse.

1.5 Research Questions

During the early stages of this research four initial research questions were formulated about investigations on the area of the GoF design patterns:

Question-1: *"What are the claims that are made about using design patterns?"* For the purpose of using design patterns a group of claims has been extracted from certain literature. They specified the functions of using design patterns. This can be used to identify the evidences in terms of where more primary studies are needed as a guide to further research.

Question-2: *"Which of the GoF patterns have been evaluated empirically?"* It helped us to identify the most frequent ways of using design patterns and their purposes during application in software engineering.

Question-3: *"For the GoF patterns that have been evaluated, what lessons about their use, and any consequences of their use, particularly regarding maintenance, are available from the empirical studies?"* Through this question we can identify what evidence currently we had and support extracting information about design patterns reuse for the following studies.

Question-4: *"What further research, using which forms, might be needed to address any 'gaps' in the available evidence?"* It helped us to decide which form of study we need to investigate in the further research.

These four questions were the very basic ones and they motivated us to investigate the area of design patterns comprehensively and deeply. Three of the four initial questions (2-4) then were addressed by means of a mapping study. This mapping study brought us quite comprehensive data about the area of design patterns.

After performing the mapping study, we found rather mixed evidence about the scope of usefulness for the patterns studied, and also there was little evidence in the form of surveys. In particular it was clear that generic claims about the value

of design patterns were inappropriate and that each pattern should be assessed separately to determine its usefulness to different groups and in different phases of software development. As the next step, we therefore sought to find out more about which patterns from the set in the GoF text were perceived to be of value and hence would be likely to be used by developers and also which ones were little valued or used. For this 'follow-on' study we therefore adopted the additional research question:

Question-5: *"Which design patterns from the GoF, do expert pattern users consider as useful or not useful for software development and maintenance, and why?"* This question helps us to investigate the developers and maintainers' views for exploratory purpose and explanatory purpose.

To address this, we conducted an online survey of experienced pattern users which were extracted from the mapping study.

The mapping study and the online survey brought us a variety of forms of evidence for the GoF design patterns. We also identified some claims about design patterns for answering Question-4. This evidence collected the different views of pattern users on the experiences of using design patterns. Therefore for the final stage, we synthesised all of the evidence to find whether there are consistencies or differences between them. For this final study of evidence synthesis we therefore adopted the following research questions:

Question-6: *"How well does the available empirical data support the claims that are made for design patterns?"* By identifying the claims we can investigate how thoroughly the evidences are addressed by the various studies.

Question-7: *"Which patterns have been studied most widely?"* It helps us to identify which patterns are considered useful and which pattern are considered less useful.

Question-8: *"How does the use of design patterns influences the process of software development?"* This is the main question of our research. It helps us to identify the factors between using design patterns and software development.

Before performing both the mapping study and the online survey, we create a protocol for each of them (Kitchenham & Charters, 2007). The protocols are provided in Appendixes. A key element of the protocol is the research questions that the study sets out to address.

1.6 Contribution of This Thesis

From our primary study it is very clear that the most of the current literature about patterns are just concerned with *how to apply design patterns* (Dasiewicz, 2005; Meister *et al.*, 2004; Ng & Cheung, 2005; Riehle, 1997; Schmidt, 1997), *how to write patterns* (Decyk & Gardner, 2007; Heer & Agrawala, 2006; Kennedy, 2004; Meszaros & Doble, 1997; Thu & Tran, 2007), *how to find patterns* (Kambayashi & Ohki, 2003; Kim & Khawand, 2007; Lee *et al.*, 2007; Martin, 1995; Wenzel, 2005), and *how to build tools for using patterns* (Florijn & van Winsen, 1997; Mak *et al.*, 2004; Mapelsden *et al.*, 2002; Noda & Kishi, 2001; Riehle, 1996). Most of them assume that design patterns are effective, but there is often no comprehensive and systematic investigation about the design patterns. And also some studies have produced conflicting results when they replicated earlier work (Prechelt *et al.*, 2001; Vokáč *et al.*, 2004). But we never know about what claims and factors influence the results.

Therefore the contributions of this thesis to the application of the design patterns in software development has three elements.

- Firstly, the claims and factors that are considered to influence design pattern reuse in software development. We have investigated the description in the literature about the functions of using design patterns and why people consider them as positive.

- Secondly, the comprehensive investigation of the area of design patterns. We have investigated the evidences to find that what evidence exists and what 'gaps' in evidence exist for reusing design patterns.
- Thirdly, the assessment for the specific patterns, and the validation of the claims and factors which are considered can influence patterns application. Our results of evaluation and validation can create a clearer view of using design patterns. Through the results we can have more evidence for the effectiveness of specific patterns. And we also can identify the key claims and factors and how well they influence pattern use in real practice.

These three contributions in these thesis will improve the evidences for design patterns application, and will help people to know the properties of design patterns clearly, so that can help them reuse more efficient in their systems creation.

1.7 Criteria for Success

In evaluating the successfulness of this thesis, the following criteria were established. Data from the mapping study, the online survey and the previous studies can be used to assess:

1. Address and identify the problems and research gaps for design patterns;
2. Examine the effectiveness of the survey;
3. Agreement or disagreement between the mapping study data, the survey data and the previous studies;
4. Assess the effectiveness of the set of patterns provided by the GoF.

Evaluation of these criteria will be analysed in Chapter 9.

1.8 Thesis Overview

This thesis is organised into nine chapters. The remaining chapters are briefly described as followed:

- Chapter 2 describes the sources of design pattern, the forms of design patterns, and what the areas the design patterns apply to. Also the claims about design patterns are also described.
- Chapter 3 describes the research methods used, describes the rationale for choosing these forms, the plans and how these are used.
- Chapter 4 describes the process of performing the mapping study.
- Chapter 5 introduces the outcomes from the mapping study in terms of the empirical papers found and their conclusions.
- Chapter 6 focuses on the process of online survey as used to investigate the experiences of software developers.
- Chapter 7 analyses and describes the results generated from the survey.
- Chapter 8 synthesises the evidence generated from the mapping study, the online survey and the claims, and then analyses the results and the threats to validity.
- Chapter 9 concludes the research results and restates the contributions, and also describes the future work.

Chapter 2

PATTERNS BACKGROUND

2.1 Introduction

According to the description in Longman's dictionary (Summers, 2006), 'pattern' when used as a noun is described as having three characteristics.

- First of all it is based on something that has happened. Generally speaking it describes a recurring thing which has occurred several times.
- Secondly, the recurring thing could be an arrangement of shapes, sounds, words or people's ideas that has common points and could be generalised.
- Thirdly, the generalised recurring thing could be repeated, copied and transferred between similar situations that follow.

Patterns appear in different areas. A pattern is a kind of abstract format for reuse that helps to reduce complexity (Pree, 1995). Within the discipline of software engineering, the idea of a pattern is employed in a wide variety of domains such as graphical user interfaces, databases, and distributed communication software (Gamma *et al.*, 1995). These patterns aim to accomplish different tasks. For example, a graphical user interface pattern makes it easier for users to understand an interface, while a database pattern is used to create databases more easily.

No matter which form of pattern it is, each of them satisfies most of the characteristics listed above. These patterns are valuable problem solutions based on

quantitative validation, and they are derived from successful practice experiences (Schmidt *et al.*, 1996). Therefore good mature patterns can improve the communication of ideas between designers.

Like most of the patterns described above, a design pattern involves abstract and concrete objects and interactions (Sommerville, 2007). It is relevant to both the software development and maintenance processes. With the publication of the milestone textbook called 'Gang of Four' (Gamma *et al.*, 1995), often abbreviated to *GoF*, the 23 design patterns in the GoF have became the most well-known patterns in Object Oriented software development. These design patterns can be viewed as abstractions for reusing ideas for components based on the object oriented design characteristics such as abstraction, encapsulation, polymorphism, and inheritance (Freeman *et al.*, 2004; Sommerville, 2007). In this thesis the research described concentrates on the 23 design patterns catalogued in the GoF.

The objectives of this chapter are to introduce the evolution and history of design patterns, to explain how design patterns have been described in the *GoF* and how they have been interpreted in the context of OO and the GoF. And it also introduces the concept of claims for patterns and the expected effects of using design patterns.

2.2 History of Design Patterns

2.2.1 Source of Contemporary Pattern

The word 'pattern' originates from the French 'patron'. Initially, it had the same meanings as the word 'patron', but it had another meaning that someone paid for the work to be done and provided an example for the worker to copy. Then the word 'pattern' separated from the word 'patron' in 16th century, and the word 'pattern' came to represent the meaning of 'example' and 'exemplar' (Onions *et al.*, 1966). It has also contained the meaning of decorative design from that time.

In the 14th century a pattern was seen as an object serving as a model or specimen (Onions *et al.*, 1966). From the very beginning the concept of a pattern was applied to different areas. For example, a French tailor produced a pattern for others to use. A hundred years later patterns became more and more important for the tailoring business.

As described above, the concept of a pattern appears in many domains nowadays. One contemporary use of patterns is taken from the area of architecture. Between the 1960s and the 1970s the architect Christopher Alexander wrote several books about urban planning. In 1964, Alexander published a book *Notes on the Synthesis of Form* (1964) that was considered as one of the most important contemporary books in the art of design. In this book Alexander pointed that contemporary design methods failed to satisfy the requirements of the individual and society (Lea, 1994). Then in 1975 Alexander published another book *The Oregon Experiment* (1975) that described an experimental approach. This book resulted in the two best known books A *Pattern Language: Towns, Buildings, Construction* (1977) and *The Timeless Way of Building* (1979) which introduced the concept of 'pattern' in the area of architecture . In these books Alexander wanted to improve people's living quality through creating better structures. In the book *The Timeless Way of Building* Alexander especially described a variety of patterns in space, human existing, events, and etc (Lea, 1994).

2.2.2 Development of Patterns in Computer Science

Although the books of Alexander were concerned with creating and using patterns in the architecture domain, they influenced the domain of computer science very deeply in the following decades. With respect to the domain of computer science, the design pattern concept can be applied in the areas of software design, configuration management, user interface design and interaction scenarios (Sommerville, 2007).

For the domain of software engineering, interest in the use of design patterns began in the 1980's. In 1984 Donald Knuth invented a programming approach

that was termed 'Literate programming'. Actually patterns were a kind of literate form which was used to solve some common but difficult problems (Buschmann *et al.*, 1996). In 1988, Ward Cunningham and Kent Beck (Beck, 1988) developed elegant user interfaces in Smalltalk by using some ideas from Alexander's architecture patterns. They created five patterns as guidelines for using Smalltalk. After that, Jim Coplien developed a set of patterns called *idioms* in C++ (Coplien, 1991). These idioms were a kind of low level pattern specific to a programming language (Buschmann *et al.*, 1996). And from the 1990s, the interest in patterns was increasing. Meanwhile, Erich Gamma started to pay attention to recurring structures and patterns in his PhD thesis (Gamma, 1991). In 1991, many professionals including Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides, (who would later be the members of the 'Gang of Four'), gathered together to discuss about patterns in a workshop at OOPSLA'91 that was organised by Bruce Anderson (Anderson, 1992).

In 1993, the first version of a catalog of patterns was published by five people including the GoF (Gamma *et al.*, 1993). This would became the basis for the milestone textbook which would published two years later.

2.2.3 The Emergence of Software Design Patterns

In 1993, Kent Beck and Grady Booch sponsored a mountain retreat for a group meeting to discuss for applying the ideas of Alexander in software patterns, and agreed to work on the basis of Erich Gamma's set of Object-Oriented patterns. That group would later be known as 'Hillside Group'[1].

Eventually in 1994, there were two events that were seen as significant for software patterns. The one was that the first flagship *Pattern Languages of Programs* (PLoP) conference was held at the Allerton Park estate near Monticello, Illinois. The PLoP conference was planned to be as an annual conference to promote implementation and research on software design patterns and pattern languages. The first conference proceedings from this was published in the following year

[1] http://hillside.net/

(Coplien & Schmidt, 1995). The other major event was that the 'Gang of Four' unveiled their milestone textbook *Design Patterns: Elements of Reusable Object-Oriented Software* (1995) at the Ninth Annual Conference on Object-Oriented Programming Systems, Languages, and Applications (OOPSLA'94). This book has been seen as one of the most influential books in software engineering design, and a milestone for the development of design patterns in the following years.

Starting with the textbook by the 'Gang of Four' (Gamma *et al.*, 1995), the literature on design patterns and their use in software design has become quite extensive. There are several notable books about software patterns based on the concept of GoF that emerged in the following decade. For example, The series of books about 'Pattern-Oriented Software Architecture' (Buschmann *et al.*, 1996, 2007; Buschmann & Schmidt, 2007; Kircher & Jain, 2004; Schmidt *et al.*, 2000) focus on research into how patterns can be used for software architecture.

2.3 How Patterns Are Encapsulated

According to the definitions of Alexander (1977) and later of Sommerville (2007), design patterns describe recurring problems and their successful solutions with a high level of abstraction and they are a fundamental means of design reuse in Object-Oriented development. From the definition, design patterns contain four essential elements (Gamma *et al.*, 1995):

- *Pattern Name.* This summarises the problem, solution and consequences of a design pattern briefly. And also it can be used to provide a common vocabulary between designers and maintainers.
- *Problem.* Describes and explains what kind of problem the pattern aims to solve.
- *Solution.* Describes how to solve the problem of design by explaining the elements' relationships, responsibilities and collaboration.
- *Consequences.* Describes the results of using pattern, and also concerns the possible impact on a system' properties such as flexibility, extensibility.

Under the four essential elements, design patterns also follow a standard format of contents. Each essential element contains some contents to make it more specific. Gamma *et al.* (1995) created a template for these contents. Table 2.1 summarises the four essential elements and their contents. The ordering of the contents is slightly different with the format used in (Gamma *et al.*, 1995) for describing the relation between the elements and contents conveniently. The contents cover all

Elements	**Contents**
Name	Name and Classification Also known as
Problem	Intent Motivation Applicability
Solution	Structure Participants Collaboration
Consequence	Consequences Implementation Sample Code Known Uses Related Patterns

Table 2.1: Essential Elements and Contents of Design Patterns

the functions of the four essential elements and are intended make design patterns clearer, more understandable to users. The format for a design pattern contains 13 contents (Gamma *et al.*, 1995):

- *Name and Classification.* It defines the pattern's name as a common vocabulary for designers and maintainers. The pattern is also classified based on purpose and scope criteria which are described in the next paragraph.
- *Also known as.* Provides an alternative name for the pattern.
- *Intent.* Describe the intention of using a pattern, and the general method applied.

- *Motivation.* Describes what problem motivates a user to use the pattern.
- *Applicability.* Helps users to known what situation they should apply the pattern, and how to recognise this situation.
- *Structure.* Presents the relation between objects in the pattern by using diagrams.
- *Participants.* Describes the classes and/or objects which participate in the application of the pattern.
- *Collaboration.* Describes the collaborations of the classes and objects to perform the pattern application.
- *Consequences.* Describes the result of using the pattern, and its possible properties.
- *Implementation.* Describes what aspects the user should be aware of in the design process.
- *Sample Code.* Provides some sample code fragments for the design.
- *Known Uses.* Illustrates some examples taken from the real systems.
- *Related Patterns.* Describes the relations with other patterns.

As mentioned above, the GoF design patterns are also classified based on purpose and scope (Gamma *et al.*, 1995). The purpose criterion classifies the design patterns into *Creational*, *Structural* and *Behavioral*. *Creational* focuses on the object creation process. *Structural* concerns classes and objects composition. *Behavioral* focuses on the relation between classes and objects. The purpose criteria is used to classify the design patterns according to whether the pattern is applied in classes or objects. Based on the purpose classification, the 23 design pattern of the GoF textbook are described in three classifications. And also these patterns are referenced by a model of 'name + page number' relating to their occurrence in this book.

2.4 How it works

"Patterns are a means of documentation." (van Vliet, 2000)

Design patterns are not specific methods as libraries and frameworks. They provide something much more like guidance for software design and maintenance. However libraries and frameworks focus on the specific application on programming, and they do not consider such design properties as understandability, flexibility and maintainability. Design patterns do not focus on coding. They aim to provide access to successful solutions when software developers meet recurring problems. The structure of design patterns focuses on the relation of classes and objects rather than the specific code.

Figure 2.1 describes how users employ design patterns. First of all design patterns are the documentation for successful software design in the minds of developers and maintainers. They are not specific code for programming reuse. When developers meet a design problem, they think about how to choose suitable patterns under that situation. Then the thinking of the patterns and their structures can be used in the specific programming process. The application of design patterns also make the implementation more maintainable. Design patterns provide the common vocabulary to improve the communication between designers and maintainers.

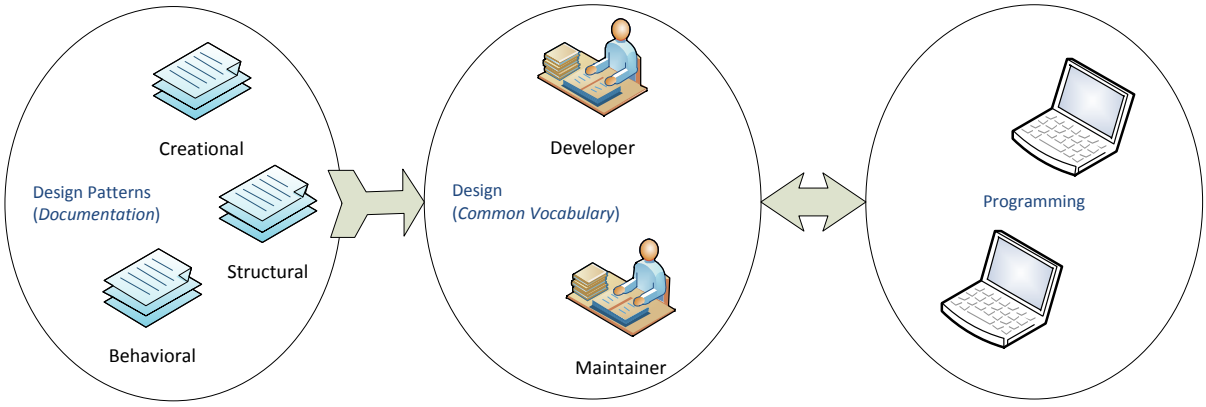

Figure 2.1: The Working Process of Design Patterns to Users

2.5 Claims Survey for Design Patterns

For the purposes of this study we set out to identify some of the specific claims that have been made about design patterns. To make this process manageable, we confined ourselves to four textbooks: the 'standard' work on patterns by the 'Gang of Four' (Gamma *et al.*, 1995); the more vital textbook for beginners in design patterns (Freeman *et al.*, 2004); the rather more architecturally focused textbook by Buschmann et al. that is widely cited in the patterns literature (Buschmann *et al.*, 1996); and the more general software engineering textbook by Van Vliet that contains a quite detailed exposition about design patterns (van Vliet, 2000). We also checked the 'Tutorial' papers which are described in Chapters 4 and 5 to find any claims about design patterns. In addition, one of the papers we analysed in Chapter 5 (Prechelt *et al.*, 2002) has identified a set of advantages as ones that are claimed for design patterns, although it does not cite specific sources for these.

We have identified twelve clear claims about patterns and their use. The claims are summarised in Table 2.2, together with our assessment of the role for each pattern implied by the claim and the label that we have used when using the claim for classification. One problem that we can identify from this exercise is that many of the claims are also stating what a pattern is or what it provides, rather than specifically stating how it impacts upon the software design process. In the next part of this section we examine each of the claims in turn and consider how its claims might be assessed empirically and the challenges involved in this.

2.5.1 Claim: Patterns encourage best practices, even for experienced designers

This claim comes from (Prechelt *et al.*, 2002) and Chapter 1, page 2 of (Gamma *et al.*, 1995). Gamma *et al.* (1995) indicates that patterns help users choose design alternatives that make a system reusable and avoid alternatives that compromise reusability, which presents the same meaning of encouraging best practices. For the purpose of assessing the claim we encounter the difficulties of determining the

No.	Label	Claims & Factors
1	Encourage best practices	"Patterns encourage best practices, even for experienced designers." (Prechelt *et al.*, 2002) "Design patterns help you choose design alternatives that make a system reusable and avoid alternatives that compromise reusability." (Gamma *et al.*, 1995)
2	Vocabulary	"Design patterns improve communication, both among developers and from developers to maintainers."(Prechelt *et al.*, 2002) "Patterns provide a common vocabulary and understanding for design principles."(Buschmann *et al.*, 1996)
3	Level of abstraction	"The benefits of design patterns include the reuse of design instead of program" (Dong *et al.*, 2007) "Design patterns don't go directly into your code, they first go into your brain." (Freeman *et al.*, 2004)
4	Life cycle	"Using these patterns early in the life of a design prevents later refactoring." (Gamma *et al.*, 1995)
5	Reuse design & architecture	"Design patterns make it easier to reuse successful designs and architectures." (Gamma *et al.*, 1995)
6	Improve productivity & quality	"Using patterns improves programmer productivity and program quality." (Prechelt *et al.*, 2002)
7	Increase skills of novices	"Novices can increase their design skills significantly by studying and applying patterns." (Prechelt *et al.*, 2002)
8	Properties	"Patterns support the construction of software with defined properties." (van Vliet, 2000)
9	Software type	"If a pattern can be found in some - say two or three - commercially or otherwise widely deployed systems, it is considered to be proven." (Kerth & Cunningham, 1997)
10	Abstraction	"Design patterns help you identify less-obvious abstracts and the objects that can capture them." (Gamma *et al.*, 1995)
11	Interface	"Design patterns help you define interfaces by identifying their key elements and the kinds of data that get sent across an interface." (Gamma *et al.*, 1995)
12	Documentation	"Patterns are a means of documentation." (van Vliet, 2000)

Table 2.2: Claims for Design Patterns

qualities in a design that relate to its reusability.

Evaluating design alternatives is a critical step to choosing an appropriate design pattern. The basis of this concept is to understand the costs and benefits of using different design patterns and choose the suitable one (Pree, 1995). Empirically, this might be addressed through a study where participants were asked to decide between solutions that use different sets of patterns or other forms to provide the same architecture/design. Performance prediction could possibly provide a suitable measure (Pree, 1995).

2.5.2 Claim: Design patterns improve communication, both among developers and from developers to maintainers

The sources for this claim are from (Prechelt *et al.*, 2002) and Chapter 1, page 5 of the book by Buschmann *et al.* (1996), which discusses both architectural and design patterns. Similar to the previous claim, Buschmann *et al.* (1996) indicate that patterns provide a common vocabulary and understanding for design principles. It also has the same meaning with improving communication. Perhaps not surprisingly, this claim is very abstract and may need to be assessed at different levels of design knowledge and experience. Two aspects that may be useful are:

- the vocabulary itself and its use for sharing ideas (what we can consider as the more 'shallow' aspects);
- its use as a vehicle for gaining an understanding of design principles - with the associated cognitive element presenting a much large empirical challenge.

For the first of these, as there are different approaches used to documenting patterns, these could perhaps be compared through some form of experiment. The second is more complex, and could possibly be approached by agreeing a set of design principles and then conducting a study where participants were asked to evaluate documented patterns against these principles.

From the points of developers and users design patterns provide a common vocabulary for developers to communicate (Freeman *et al.*, 2004). Also from the point of software design, providing comments in code can help developers identify patterns clearly. Thus for this assertion we have narrowed it down as 'Developer communication' and 'Code comments & naming convention' during extraction from the collected papers.

2.5.3 Claim: The benefits of design patterns include the reuse of design instead of program

This claim from (Dong *et al.*, 2007) indicates that design patterns are focused on design rather than programming. Freeman *et al.* (2004) presents the same thought about that, it indicates design patterns do not translate into code directly.

As described in the previous section, design patterns can be used as documentation. They are not like libraries and frameworks which are employed directly in programming. Design patterns do not instruct users how to program. They are encapsulating design concepts, and provide users with suitable problem solutions from the design aspects.

2.5.4 Claim: Using patterns early in the life of a design prevents later refactoring

This claim is extracted from Chapter 6 of textbook (Gamma *et al.*, 1995). As is widely recognised, software has its own life cycle. Once software is put into service as a mature product, here are two problems that commonly occur (Gamma *et al.*, 1995):

- more requirements
- need to be more reusable

The first point means that software should provide more functions. It requires software design to satisfy the need for extensibility. The second one means software can be reused in the process of evolution. As successful solutions, design patterns are used to make systems to satisfy these two points. Design patterns have been considered with properties such as extensibility. And also the successful reuse of design patterns should makes a system easier to maintain in the later stage of software life cycle.

2.5.5 Claim: Design patterns make it easier to reuse successful designs and architectures

This claim is taken from Chapter 1, page 2 of the textbook by the 'Gang of Four' (Gamma *et al.*, 1995). This is really a definition statement since design patterns are a means of packaging and documenting experience of successful designs and architectures. So we can consider that the main claim is centred upon the extent to which a design and architecture can be reused when packaged in this form, and what is implied by the term 'easier'. Interpretations of 'easier' could include one or more of:

- it being faster to complete the design
- the final system being less error prone
- the design could be produced by a less experienced designer

when compared against a baseline of a design produced without using patterns.

While it would be non-trivial to do so, these comparisons could be made by using controlled experiments producing solutions by using or not using design patterns; via a case study of two teams doing likewise (allowing for a larger system); or through observation, finding comparable open source solutions produced by both means and looking at (say) defect rates. The first two approaches would require independent development of the test cases to ensure equality of functionality and to compare defect rates. Managing the skill levels and experience of participants would also present a challenge.

2.5.6 Claim: Using patterns improves programmer productivity and program quality

This claim comes from the paper (Prechelt *et al.*, 2002). As noted in the definition of design patterns in the previous section, design patterns provide successful recurring solutions for software developers with some defined properties. If a programmer who has enough experience of software design and employs patterns appropriately, can get more improvements in productivity and quality than without using patterns. And the programmer's work will also be more efficient.

2.5.7 Claim: Novices can increase their design skills significantly by studying and applying patterns

This claim is also from the paper (Prechelt *et al.*, 2002). Design patterns are created based on experts' successful experiences. For novices in software design, it is obvious that they can help with certain problems in design. Therefore learning design skills through design patterns is potentially an efficient way for novices to learn from the successful experiences of experts.

2.5.8 Claim: Patterns support the construction of software with defined properties

This claim comes from the book by van Vliet (2000). Once again, there are issues of vocabulary here, in terms of what exactly we consider to be a 'defined property'. In this claim, design patterns help developers to construct software meeting a skeleton of functional behaviour and non-functional requirements (Buschmann *et al.*, 1996). With the guidance of this claim, we classified the collected papers according to the concepts of 'skeleton of functional behaviour' and 'non-functional requirements'. The functional behaviour deals with the specific functions based on a system's specific requirements. The non-functional requirements are related to the functionalities such as reliability, cost, maintenance or development.

2.5.9 Claim: A design pattern is considered to be proven through two or three commercial or otherwise widely deployed systems

This claim is extracted from the paper (Kerth & Cunningham, 1997). The eventual objective of design patterns is to be successfully employed in real systems such as commercial systems. Therefore the experiences of some real system design can be used to evaluate the usefulness of design patterns. In this claim the issues of 'how to prove successful patterns reuse' and 'the differences of the system types' are key ones. The issue 'how to prove successful patterns reuse' is related to the previous claim 'reuse design & architecture'. And also we may be concerned as to whether there are differences between the effects of using patterns on different software types during the process of using patterns.

2.5.10 Claim: Design patterns help you identify less-obvious abstractions and the objects that can capture them

This is taken from Chapter 1, page 13 of (Gamma *et al.*, 1995) (from the section on 'finding appropriate objects'). Perhaps the key issue associated with this claim is of determining exactly what comprises a 'less-obvious' design solution, and particularly, in the sense of 'obvious to who'? The designer's level of knowledge is an unavoidable factor.

If we take the view that a pattern might be less obvious if it needs to be dispersed among a number of objects and functions, then it might be possible to create a measure of 'dispersion' and to then correlate this with the degree of 'maintainability' created by using the pattern.

2.5.11 Claim: Design patterns help you define interfaces

Again, this is taken from (Gamma *et al.*, 1995), Chapter 1 and page 4 (from the section on 'specifying object interfaces'). Like the claim 'reuse design & architecture', it also has a large element of definition, making it intrinsically difficult to test systematically. In OO software design process, programming to

an interface is regarded as a design principle (Freeman *et al.*, 2004). And the design patterns in GoF implement this principle.

2.5.12 Claim: Patterns are a means of documentation

The source for this claim is Chapter 10 of the book by van Vliet (2000). While appealing in terms of the implications for the wider software development community, it is again in part a definition, and like some of the other claims, there is no obvious comparator to measure against and to use as a baseline.

The interpretation sometimes used for this is that a given design could be documented in two ways: one that makes explicit where any patterns have been used and one that does not. An experiment could then involve comparing ease of change for various purposes across the two design forms.

From the technique aspects during the software evolution procedure it may be necessary to extend and modify the original architecture and modify the system's code. Used as means of documentation, design patterns can improve the documentation and maintenance of existing system, helping developers to extend and modify the architecture and code (Buschmann *et al.*, 1996)

2.5.13 Using the Claims as a framework

We derived our set of claims independently, so it was interesting to observe how far the studies that we performed in the following chapters specifically addresseed some of the claims that we have identified.

It is perhaps hardly surprising that empirical studies of design patterns rarely seem to address these claims directly - however, in studying specific, often more detailed, aspects of patterns and their use, an empirical study may address elements of one or more of these. We have therefore used them as a basic framework for categorising the studies that we found, with the particular aim of finding out how extensively each one has been assessed - even if not directly.

We should of course note that strictly, the claims should be assessed on a 'per-pattern' basis, since each pattern addresses different design issues and involves varying cognitive complexity. Indeed, two of the studies in the mapping study (Prechelt *et al.*, 2001) and (Vokáč *et al.*, 2004) do make some per-pattern assessments (and get different results for some).

2.6 Summary

In this chapter we have first described the history of design patterns as a concern. The OO software design patterns become popular in software design from 1995 when the milestone textbook GoF was published. Since then the 23 design patterns in the GoF were employed by developers and maintainers.

Then we also focused on the contents of design patterns, and described how they work for software design. In the GoF textbook the 23 design patterns are classified according to *purpose* and *scope* criteria. As for a single pattern, it contains 4 essential elements and consists of 13 contents. As a documentation design pattern brings the thinking of design to developers and maintainers rather than help them programming directly.

Besides that, we also performed a claim survey for design patterns and extracted 12 claims from some textbooks and papers we had investigated in the mapping study. These claims specify the functions and properties of using design patterns. The results of the claim survey can be used to help with synthesising the results of the mapping study and the online survey in Chapter 8.

Chapter 3

RESEARCH METHOD

3.1 Introduction

While the concept of 'patterns' has become popular for a number of pruposes (Buschmann *et al.*, 1996), it has been widely advocated for designing Object-Oriented software systems (Gamma *et al.*, 1995). Researchers and practitioners have become interested in finding and writing design patterns to assist with the software development and maintenance processes [1]. Since the publication of the book by the 'Gang of Four'(GoF), much of the research and the development about design patterns has also been based on the 23 design patterns described in the GoF.

But where do good design patterns come from and how effective is the concept? The good design patterns are claimed to originate from the practical experiences of software developers (Schmidt *et al.*, 1996). However, the effectiveness of design patterns, either as a concept, or individually, has not been empirically validated on any very comprehensive basis.

Therefore in order to perform a comprehensive investigation of the 23 design patterns and to fulfil the research goal described in Chapter One, a combination of empirical forms have been employed to help elicit knowledge about how

[1] www.hillside.net

experts and novices use patterns. These consist of two methods: namely a *mapping study* and *surveys*. The surveys included an *online survey of practitioners* (human-based) and a *survey of claims* (document-based). Finally a process of *evidence synthesis* is used to compare the outcomes.

This chapter aims to describe the research methods applied in this thesis. It starts from the description of the scope of the research. The overall process of the research which consists of the research methods is described. Then the following sections describe each research method applied in the research and its importance, and also describe how the research was organised.

3.2 The Scope of The Research

In the previous section it was stated that this research contained four elements. Figure 3.1 shows how these methods are involved, and their inter-relationships.

- *Mapping Study*: This was the first stage, which was concerned with identifying empirical and observational studies about design patterns and their effectiveness, and classifying the papers around specific design patterns. That provided the fundamental materials for the research of the next stages. After papers were classified the following methods could use the data extracted from these papers. (Data set A)

- *Online Survey*. The purpose of this was to augment the empirical studies by directly collecting experience about use of the GoF patterns to see if this provided further insight and any reinforcement on the application of design patterns. Experiences about using patterns were extracted from the observational experience papers found in the mapping study, and used to help create the questions for the online survey. Then a set of experts were identified as our sampling frame, and their responses formed data set B.

- *Claims*. This was a document survey to extract the claims about design patterns found in the mapping study and the books. In the mapping study all papers found in the search phases were classified. The claims were

extracted from some textbooks about design patterns and the classification category of "Tutorial", which focuses on "how to use patterns, discussing and teaching them". This survey generated the data set C.

- *Evidence Synthesis* forms the final method. This was applied to synthesise the data from A, B and C and to analyse the results from them.

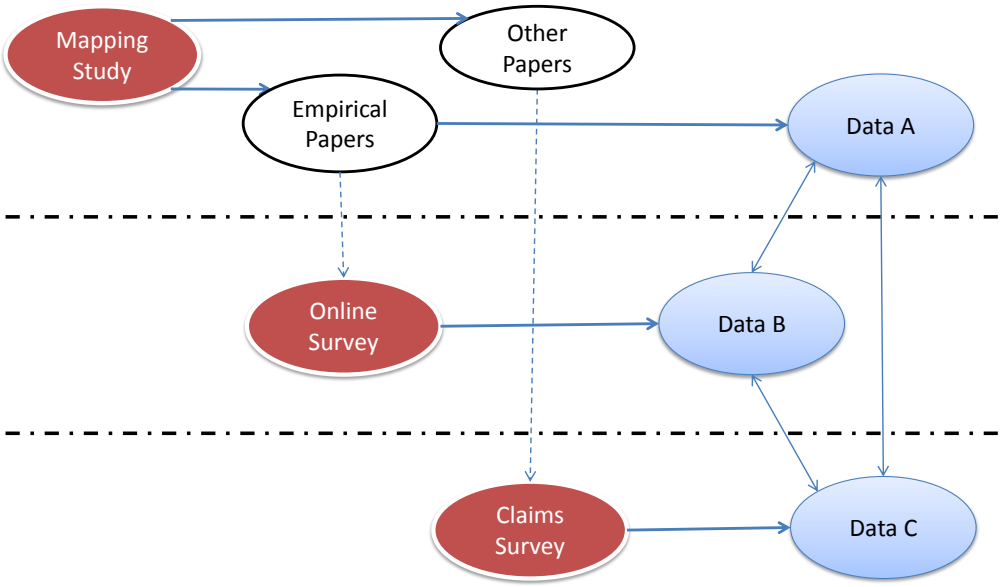

Figure 3.1: Methodological Process and Inter-relationships

Using the category model for empirical research described in (Oates, 2005), the different aspects of this research can be categorised using the 6Ps, namely: *purpose*, *products*, *process*, *participants*, *paradigm* and *presentation*.

- *Purpose*: describes the reason for doing this research, and what contributions it is expected to provide the knowledge.
- *Products*: describes the results generated from the research. The results can be used to support the "*purpose*".

- *Process*: describes how the results are to be obtained and how to achieve the purpose. The process includes details of the strategy and method application.
- *Participants*: describes these people who participate in the research in any way.
- *Paradigm*: describes the evidence and the model being extracted from the collected evidence and the generated conclusions in our research. We generate the conclusion by using the collected evidences.
- *Presentation*: aims to clearly describe and explain the other points above.

These categories cover all aspects of the research. Therefore in this chapter the 6Ps categories are applied to describe the research and to structure the discussion of each method used. We employ the 6Ps categories respectively for the mapping study, the online survey and the evidence synthesis methods, to explain why they were applied, the process of the application, the outcomes, and the positive and negative aspects. Because the description process in forms the process of presentation, so the element "*presentation*" from the 6Ps categories is not explicitly included in this chapter. Also, since the "*claims*" have been described in Chapter Two, this chapter just focuses on using these to describe the mapping study, the online survey, and the evidence synthesis.

3.3 The Mapping Study

This is an evidence-based approach widely used in software engineering research (Kitchenham *et al.*, 2010). The concept of evidence-based research was originally developed in the area of clinical medicine. With its success in clinical medicine, evidence-based research was adopted for use in other domains, including software engineering. Evidence-based software engineering aims "to provide the means by which current best evidence from research can be integrated with practical experience and human values in the decision making process regarding the development and maintenance of software" (Dybå *et al.*, 2005).

The core tool of the evidence-based paradigm is the Systematic Literature Review (SLR), often abbreviated to 'Systematic Review' (Kitchenham & Charters, 2007; Petticrew & Roberts, 2005). This provides a framework for systematically searching the literature, extracting the data, and performing the necessary analysis. A mapping study (also termed a scoping review) is sometimes used before undertaking a systematic literature review, is designed to provide a wide overview of a research area, to establish if research evidence exists on a topic, provide an indication of the quality of the evidence and identify where 'evidence gaps' and 'evidence clusters' may occur (Kitchenham & Charters, 2007). Here the research method adopted has been to make use of a mapping study to identify how extensively patterns have been studied. Table 3.1 highlights some key distinctions between a mapping study and a systematic literature review in terms of some of the main elements (Zhang & Budgen, 2011b).

3.3.1 Purpose

The mapping study was performed by following the guidelines proposed by Kitchenham & Charters (2007). From the comparison between mapping study and systematic literature review in Table 3.1 it can be seen that a mapping study is more suitable than a systematic literature review for our purpose, because before this research there was little systematic collection of evidence about design patterns and also the topic of design patterns is abstract and broad. The form of a mapping study can be applied to classify and analyse the literature about design patterns. Also the purpose of a mapping study is to identify 'evidence clusters' and 'evidence deserts' with regard to empirical studies of the chosen topic , which here is that of design patterns.

We set out to answer the specific research questions below by performing this mapping study:

- "*Which of the GoF patterns have been evaluated empirically?*"
- "*For the GoF patterns that have been evaluated, what lessons about their use, and any consequences of their use, particularly regarding maintenance, are available from the empirical studies?*"

Element	Mapping Study	SLR
Goals	Classification and thematic analysis of literature on an SE topic	Identifying best practices with respect to specific procedures, technologies, methods or tools by aggregating information from comparative studies
Research question	General - related to *research trends*. Which researchers; how much activity; what type of studies etc.	Specific - related to *outcomes* of empirical studies. Of the form: "does technology/method A have property X?"
Search process	Defined by the topic area.	Defined by the research question.
Required search outcomes	Less stringent if only research trends are of interest	Extremely stringent - all relevant studies need to be identified.
Quality evaluation	Not essential	Important to ensure that the results are based on best quality evidence.
Results	Set of papers related to a topic area, categories for these, and counts of papers in each category.	Answer to specific research question, possibly with qualifiers (e.g. that results apply to novices only).

Table 3.1: Differences between Mapping Studies and SLRs

- "*What further research, using which forms, might be needed to address any 'gaps' in the available evidence?*"

Answers to these questions were expected to provide the systematic evidence for planning further research about design patterns. The results of these questions display an overview of the field of design patterns, making clear where evidence

is lacking and what needs to be investigated in the future research.

3.3.2 Products

The expected outcome of the mapping study was the collected literature about the GoF design patterns. The papers found by the search were classified based on the form of study and the issue about design patterns that they addressed. The details of the classification are described in Chapter 4.

In the classification about the form of study a number of papers were classified as 'empirical'. After further selection for relevance the mapping study analysed the papers describing controlled experiments. It then also examined the observational 'experience' reports describing application of patterns and used some of these for supplementary evidence. (There were no available case studies and surveys.)

Through analysing this literature, the outcomes helped to form a clear framework about design patterns, and also provided the basis for both the organisation the online survey as well as an input to the final results of this thesis.

3.3.3 Process

In terms of the description provided in Kitchenham's guidelines (Kitchenham & Charters, 2007), the mapping study was divided into three main phases, namely *Planning the Review*, *Conducting the Review*, *Reporting the Review*. The details of the three phases are described in Chapter 4 and Chapter 5. Here we describe the main features of each phase and their motivation.

- *Planning the Mapping Study.*
  - **Identification of the need for the mapping study**.
    The purpose of this research and the mapping study have been described in the previous chapters and sections. The analysis of its outcomes should make the research needs and the research questions clearer. Before performing this research about design patterns, there was little systematic evidence about the effectiveness of GoF design

patterns. And also the concept of a design pattern is broad in scope and abstract. For instance, one design pattern might be helpful to a experienced developer from one angle, but it might not be helpful to a different group of researchers and developers. Therefore we aimed to investigate the value and scope of design patterns by performing this mapping study.

- **Specifying the research questions**.
  In the previous section the research questions for the mapping study have been outlined. These research questions were addressed by performing the mapping study, and also the results of the mapping study were extracted to answer these questions. So these questions were chosen to assess the evaluation and effectiveness of design patterns, and also to provide the baseline material for the next phase of research.

- **Developing and evaluating a protocol for the mapping study**.
  In order to specify the method applied in the mapping study, a predefined protocol was created before performing the mapping study. The protocol defined a series of specific issues for the mapping study, that included *Background*, *Research Questions*, *Search Strategy*, *Study Selection Criteria and Procedures*, and *Data Synthesis*. After producing the protocol, it was evaluated by the supervisor. The details of the protocol are shown in Appendix A.

- *Conducting the Mapping Study.*

The protocol for the mapping study formed the basis for conducting it. The mapping study was performed following each step of the protocol, with the details of this being described in Chapter 4.

Figure 3.2 summarises the process of the mapping study. The mapping study was organised around three rounds of searching. The first round was

the online searching of a set of digital libraries. The second round performed a manual search of four journals. The papers found were classified into five groups based on the category *Pattern Theme*. In these five groups one group was classified as empirical one which was conducting empirical forms of study about design patterns. Then we continued to check the references of the empirical papers. This is called snowball searching. The papers collected about design patterns from this were also classified use the above the category. At last the empirical papers were also classified according to two categories *Form of Study* and *Pattern Issue*.

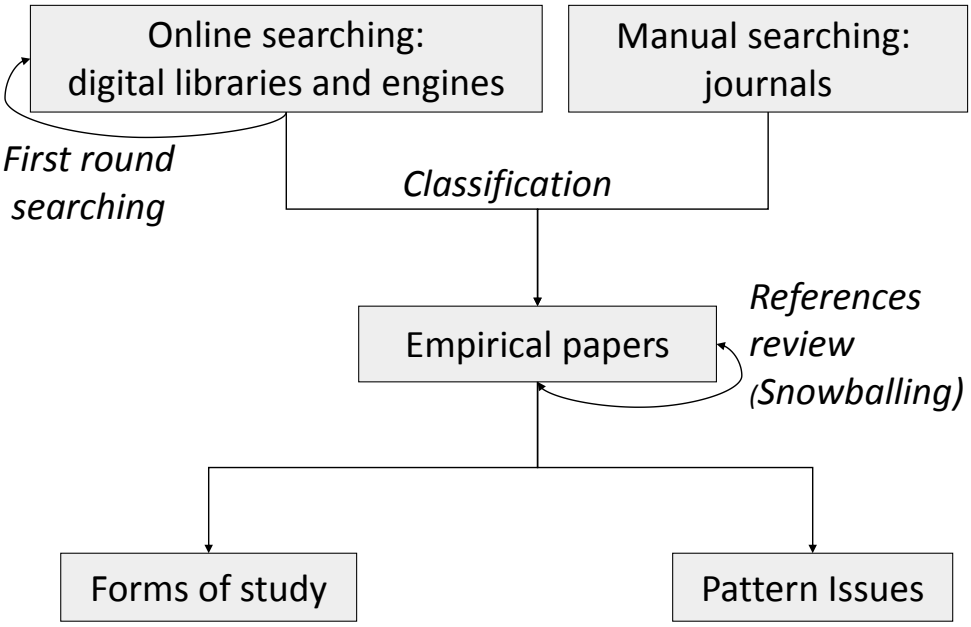

Figure 3.2: Mapping Study Flowchart

Finally, we focused on the empirical papers describing experiments and experience, to analyse their outcomes to help answer the research questions for the mapping study. The details of the analysis process and the outcomes from this are described in Chapter 5.

- *Reporting the Mapping Study.*

  The search and classification details for the mapping study are reported in Chapter 4, which describes the process of the mapping study, and Chapter 5 which provides the outcomes and analysing data of the mapping study. Also the process and outcomes have also reported in an EASE conference paper (Budgen & Zhang, 2009) and submitted to a journal (Zhang & Budgen, 2011b).

3.3.4 Participants

In the mapping study the author was the leading participant. He was directly involved in the research through the whole process. His supervisor reviewed the protocol and acted as the second analyst for the inclusion/exclusion process. Besides that, the referees who provided feedback for the EASE conference paper and journal submission can also be considered as participants.

3.3.5 Paradigm

In this mapping study we extracted the models from the collected experiment papers and the experience papers. To the experiment papers the models about their research questions, the structures of the experiments, and the patterns being studied were extracted as models for analysis. For the experience papers a learnt lesson model was created. This model concluded the which patterns being studied, what lesson learnt from the experience evidence such as forms of study, type of system, and system size.

3.4 The Online Survey

Surveys are applied widely for eliciting expertise and experience that is largely contained within a reasonably well-defined community (Fink, 2002). The definition of a survey can be described as:

> "*A system for collecting data from a population or sample at one point in time where it is assumed that there is heterogeneity in personal characteristics, attitudes, knowledge, and behaviors across the population.*" (Bourque & Fielder, 1995)

With increased use of the Internet, this has come to provide a convenient and efficient way to perform a survey. Researchers can perform an online survey either by sending emails, with the survey form as an email attachment, or by using an online web form. Using the internet is much cheaper than postal questionnaires (Oates, 2005). Because this element of research about design patterns aimed to investigate the experiences of professionals with using design patterns in the domain of software engineering, not only are there many experts in the community, but also they can be expected to be familiar with using the web. Therefore an online survey was considered to be the most suitable method for performing this element of the investigation.

According to the different purposes that survey can have, it can be classified as *exploratory*, *descriptive* or *explanatory*. In order to both identify what experience is available about each pattern and also the causal links between experience and opinion where possible, this survey involved a mix of exploratory and explanatory approaches to achieve these two purposes.

Kitchenham and Pfleeger have explored how surveys can be used to address software engineering issues, and some of the problems that need to be addressed (Kitchenham & Pfleeger, 2002a,b,c,d, 2003; Pfleeger & Kitchenham, 2001). The design of this online survey was based on the template derived from their works and provided from www.ebse.org.uk. In the following sections we try to describe to design issues including identifying (and accessing) the relevant population; using appropriate forms of sampling where the population is large; obtaining an adequate response rate; avoiding bias in the questions; etc.

3.4.1 Purpose

The evidence collected about the GoF design patterns from the mapping study was not systematic nor was it comprehensive. This therefore provided only limited evidence towards answering the overarching research question "to identify how the use of design patterns influences the process of software development". Therefore, the purpose of the online survey was to investigate the specific patterns and their usefulness as perceived by researchers and developers, drawing upon their experiences. The overall research question for the online survey was:

> "*Which design patterns from the GoF, do expert pattern users consider as useful or not useful for software development and maintenance, and why?*"

3.4.2 Products

Firstly, it was decided to conduct the survey using an online questionnaire delivered via a commercial professional survey website, www.surveymonkey.com. As described above, this questionnaire was designed based on the template from the EBSE website. The details of the design process are described in Chapter 6.

Secondly, this survey aimed to collect data that reflected the experiences of a group of researchers and developers about using design patterns. The data reflected the participants' attitude towards the 'usefulness' and 'uselessness' of the 23 GoF design patterns. Each participant was asked to choose up to three patterns that were considered useful and up to three patterns that were considered not useful, according to his/her experience of research and development.

Thirdly, the data was classified according to the different groups of respondents identified. Then the different forms of data was subjected to quantitative and qualitative analysis respectively. The details about the analysing process and the outcomes are described in Chapter 7.

3.4.3 Process

In terms of the survey process described in Pfleeger & Kitchenham (2001), this online survey consisted of three phases, containing eight activities. The three phases were: *Survey Preparation*, *Survey Design* and *Data Processing & Analysing*. They were classified based on the main tasks in the different phases.

Among the three phases, the phase *Survey Preparation* involved a set of preparatory activities, including setting objectives and planning & scheduling for the survey. The phase *Survey Design* focused on the specific design process of the survey, including the design activity, data preparation & collect activity, and pilot testing. Pilot testing was employed to validating the instrument, following which, the survey design was improved based upon the pilot testing feedback. The final phase, *Data Processing & Analysing*, involved the data processing activity which was employed to validate the confidence in the collected data, and the analysing activity to analyse the data. Figure 3.3 displays the process of the online survey. The details about the outcomes and the analysing results of the online survey are reported in Chapter 7. Here we describe the main features of each activity.

- *Setting Objectives.*

  The purpose of this online survey has been described in the previous section. Therefore the research questions of this research and the online survey are set to be the objectives. They aim to assess the effectiveness of each of the design patterns in the GoF.

- *Planning and Scheduling the Survey.*

  After setting the objectives, we planned and scheduled the survey. Firstly the observations extracted from the mapping study were applied in the survey design. A survey protocol was produced based upon the template in www.ebse.org.uk and the methods developed in a survey series by Kitchenham and Pfleeger (Kitchenham & Pfleeger, 2002a,b,c,d, 2003; Pfleeger &

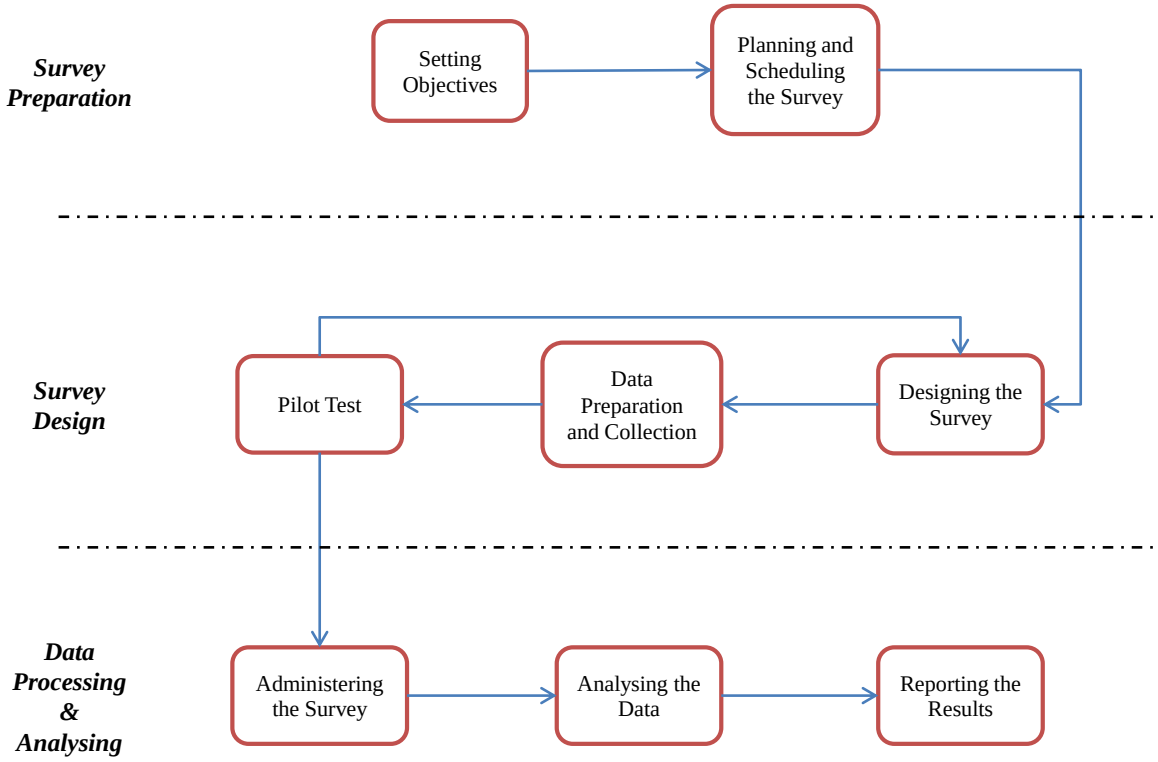

Figure 3.3: Online Survey Flowchart

Kitchenham, 2001). The purpose of the survey protocol was to summarise the background of the research about design patterns and provide a outline of the online survey. It assisted in ensuring consistent data collection during the process. This survey protocol was organised into seven parts, including:

- **Change record**. This provided a record of changes and summarising the main updates and changes for each version of the survey protocol.
- **Background**. Described the background of the research and its motivations.
- **Design**. Described the preparation tasks for the survey design.
- **Data Preparation and Collection**. Described the material required in the survey design and how to collect the data.

- **Analysis**. Briefly described the methods applied in the data analysis after collection, and used to validate the data.
- **Reporting**. Briefly described how to display the data.
- **Schedule**. Provided a general schedule for the online survey.

The details of the survey protocol are shown in Appendix B.

- *Designing the Survey.*

The design of the online survey was motivated by the research questions. The design process focused on the factors, such as the survey form, data requirements & collection, population and sampling technique, etc. These factors are illustrated below.

- **Form of survey**.
  Our survey was considered to be a mix of exploratory (identifying what experience was available about each pattern) and also explanatory (since we wanted to identify the causal links between experience and opinion where possible).
- **Data requirements**.
  The required data for the survey design was extracted from the experience papers in the mapping study. The data that is shown in Chapter 5.
- **Population**.
  Determining the population to be sampled is a key element in designing a survey. For our study we ideally wanted to survey a representative set of software researchers and developers (including maintainers) who possess experience of using patterns (and maintaining systems developed with their use). However, identifying this group and accessing them presented a major problem - since there is no obvious forum through which they could be identified.
  The solution adopted for this problem was to use a 'surrogate' group

that we were able to access, namely the set of authors of papers about patterns. Conducting our mapping study had provided us with a comprehensive list of papers about design patterns, and while we had only included these describing empirical studies in our final set, we did have access to the others (and had categorised these after excluding papers that had nothing to do with software design patterns). The set of all authors of these included papers were defined as the population.

- **Selection of participants**.
  There were two possible sources used in inviting people to participate in this online survey. Firstly the authors in all papers of the mapping study were invited. By extracting the names of the authors from these papers, we were left with 882 names after removing duplicates, and we also had details of e-mail contacts for them. Later, we also joined in a Internet technique community LinkedIn www.linkedin.com to invite the experts who are interested in design patterns to participate the survey.
- **Sampling technique**.
  As a consequence, we therefore used a mix of sampling forms. The original group of authors formed the basis for cluster-based sampling; where they passed the request to colleagues this created a snowball sample; and finally the responses from the research-oriented maillists formed a self-selection sample.
- **Sample size**.
  Since the 'normal' level of response to surveys is of the order of 10%, we hoped to obtain 70-100 responses out of the 882 authors.
- **How to collect the data**.
  Since our contact mechanism was through electronic means (e-mail) we decided to use on-line data collection through a web-based form, on the basis that this would be familiar to all involved.

- *Data Preparation and Collection.*

We investigated the experience papers generated from our literature review to extract information about a certain set of design patterns. A group of the related questions for investigating useful and useless patterns were asked to the participants.

Usually sending a questionnaire to participants should be combined with a brief request letter (Dillman, 1999). The request letter aims to describe our research purpose, the online questionnaire web address, and also the survey closing date. The request letter is shown in Appendix C (dates were changed according to the sending group).

After sending the questionnaire we started to collect the responses until we had enough to satisfy the confidence level. The responses were coded according to the receive time. The data was stored in the survey website database and also stored in our local file store for backup and further analysis.

To obtain answer to the questions in our questionnaire, we had to ensure that people were able and willing to answer these questions. In a survey it is common that some participants are difficult to contact, and also it is possible that they are reluctant to participate (Biemer & Lyberg, 2003). In order to not to miss any expert it was necessary to send reminders to the participants who do not respond to our first request. If we got a low response rate we can plan sending reminders to participants and individuals if necessary (Biemer & Lyberg, 2003; Kitchenham & Pfleeger, 2002a). But in this survey due to the high effectiveness and the low cost of email, the reminders were also sent via email rather than the traditional mail. It can help us to assess the reason of low response. The reminder letter is shown in Appendix D (dates were changed according to the sending group).

- *Pilot Test.*

A pilot test is an important part of a survey. It ensures the high quality of a survey (Wright & Marsden, 2010). The pilot test was a dry run before conducting the final version (Fink, 2002). The questionnaire form was reviewed by a small team of assessors with a questionnaire for this pilot testing. The form of this pilot testing questionnaire was based on the checklist in Fink's handbook (Fink, 2002). The questionnaire asked the assessors nine questions about whether the length of the questionnaire was suitable, whether the structure and the wording was clear and easy to understand, and whether the question choices were exhaustive, etc. Following the comments from the assessors, we made a number of changes to improve presentation and clarity. The questionnaire used for the pilot testing is shown in Appendix E.

- *Administering the Survey.*

  In order to keep the process under control we sent our requests out in batches (usually 50), following up each batch two weeks later with a short reminder message to those who had not responded. Our invitation to participate in the survey also asked recipients to pass it on to others who might be interested, which a number did do. We also sent our invitation to three research-oriented mail-groups in LinkedIn as described above. The three groups include those are interested in design patterns and Object-Oriented software, such as Design Pattern in C++, Learning Design Patterns and OOAD, and Software Design Patterns And Architecture.

- *Analysing the Data.*

  At the beginning of the analysis process, there were two questions we had to answer. The first one is that whether the response data were recorded accurately, and the second one is that whether the grouped data were classified consistently and accurately (Diamond, 2000).

We preferred a statistical analysis and classification of data from the demographic questions, so that we could be familiar the profile of the responses. Then we also did the statistical and classification analysis for the specific patterns which were considered useful and not useful.

Finally, combined with the demographic data, the data of the specific patterns were analysed. To analyse numerical data we applied the statistical forms described in (Kitchenham & Pfleeger, 2003). To analyse ordinal data we converted data to numerical values. To analyse nominal data we determined the proportion of responses in each category (Kitchenham & Pfleeger, 2003).

As with any empirical study, it is necessary to consider the possible effects of any factors that could have biassed the outcomes from the study. Here we have addressed two of these: the design of the survey instrument; and the extent to which we can assess whether our actual sample was representative of the target population.

- *Reporting the Results.*

  After the online questionnaire was closed and the data has been analysed, the results of the survey can be reported. According to the conclusion in the survey research guide (Diamond, 2000), our report should include the characteristics as following:

  - The online survey purpose;
  - The definition of the target population and our sampling technique;
  - The description of the questionnaire design process and the motivation;
  - The description of the pilot testing;
  - The description of the analysing process and the results;

– The exact form of the survey protocol, the request letter, the reminder letter, the pilot testing questionnaire, and the online questionnaire form. (These are presented in Appendixes.)

For the data analysis, we endeavored to describe the outcome data by charts and tables. The charts and tables could display the results of the analysis clearly and directly.

The details of the design process and the pilot testing are described in Chapter 5, while the details of the analysis process and the results are described in Chapter 6. In addition, we are preparing a journal paper to present the process and results of the online survey (Zhang & Budgen, 2011a).

3.4.4 Participants

As for the mapping study, the first participants should be the author and his supervisor. The author created the questionnaire, and his supervisor evaluated the products through the whole process.

The second group of participants were the assessors in the pilot testing. The assessors participated in the online survey and provided comments that were used for improving the questionnaire.

The third group of participants were the authors whose names were extracted from the collected papers in the mapping study. These people were invited to participate the survey through sending the email request letter.

The fourth group of participants were the people who was invited by the authors. The authors forwarded the request letters to the people who was considered to be interested in the area of design patterns.

The fifth group of participants were the people who were from the online community groups accessed via LinkedIn. The request letters were sent to three groups

in LinkedIn. The members of these groups were expected to be interested enough in the topic for them to be willing to participate the online survey.

3.4.5 Paradigm

In our survey we borrowed the way of thinking from the mapping study, and tried to find the models from the respondents. As with the mapping study we used models like the system type, system size. And all of the respondents' feedbacks were considered as being the evidence that would provide the models.

3.5 Evidence Synthesis

As described in Figure 3.1, there is a need to perform some forms of data synthesis between the results of the mapping study, the online survey and the claims. In the final stage of this research, the results of the synthesis can improve the quality of the answers to the research questions.

Research synthesis is a family of methods for summarising data, integrating, combining, comparing, and finding the results between the different studies on a same research topic (Cooper, 2009; Cruzes & Dybå, 2010; Dixon-Woods *et al.*, 2005; Ogawa & Malen, 1991). It provides a series of methods for synthesising the past empirical researches. So before performing the research synthesis, it is necessary to select an appropriate method for synthesis. Among these methods meta-analysis is regarded as the best choice of quantitative aggregation, which is a statistical method for data synthesis. But it has to satisfy some pre-assumptions. The primary studies for synthesis should be similar and the collected data should be in the form of quantitative data (Cruzes & Dybå, 2010). But actually most of the research in software engineering is heterogeneous, and it has been shown that it is difficult to apply this in software engineering (Brooks, 1997; Ciolkowski, 2009; Ciolkowski & Münch, 2005; Hayes, 1999; Miller, 2000). Also the forms of our studies, such as the mapping study, the online survey and the claim survey, are quite different. The results of them also consist of quantitative and qualitative. Therefore meta-analysis is unsuitable for our data synthesis.

Currently the most common method being applied in software engineering is the narrative synthesis (Ciolkowski, 2009), and it is regarded as the 'second best' approach in the synthesis of multiple studies and it can be seen as the alternative to meta-analysis (Popay *et al.*, 2006; Rodgers *et al.*, 2009). Narrative synthesis is a method for synthesising the extracted evidence with text, tables and other textual forms to interpret, compare and summarise the collected evidence from the multiple forms of studies for theory building (Cruzes & Dybå, 2010; Popay *et al.*, 2006). Therefore, the method of narrative synthesis was applied in our evidence synthesis stage to compare and summarise the evidence collected from the mapping study, the online survey and the claims survey.

3.5.1 Purpose

As described in Figure 3.1 there were three methods, the mapping study, the online survey, and the claims survey, which we had performed before this last stage. They elicited preliminary synthesis at their stages respectively. At this final stage we organised these results from the preliminary synthesis to explore the relations between them.

In the mapping study we investigated how extensively the use of software design patterns had been subjected to empirical study, and tried to find the most effective mechanism for knowledge transfer. The online survey was conducted to investigate the relations between the profiles of users and the design patterns, and also which patterns were considered as useful. The claims survey was used to found the claims about design patterns. So in this stage of evidence synthesis we organised, compared and synthesise the data from the mapping study (data set A), the online survey (data set B), and the claims survey (data set C). From the synthesis results we aimed to investigate the original questions repeated below:

- "*How well does the available empirical data support the claims that are made for design patterns?*"
- "*Which patterns have been studied most widely?*"

- "*How does the use of design patterns influences the process of software development?*"

3.5.2 Products

The mapping study, the online survey, and the claims survey had generated a preliminary synthesis of their results respective. At this stage we explored the relations between the preliminary synthesis results from them. These narrative synthesis results explained the relation between the claims and all design patterns from a general aspect, and also explained the comparison between the three results from the methods from the aspects of the specific patterns. Obviously the results of the comparison and synthesis can not be all consistent. So for the consistent results we regarded them as the certified results. Conversely the inconsistent results and the results with no effects were regarded as the uncertified results.

3.5.3 Process

According to the narrative synthesis process in (Popay *et al.*, 2006; Rodgers *et al.*, 2009), we undertook our synthesis in four steps:

- *Developing a theory.* The majority of this research aimed to investigate how the use of design patterns influences the process of software development. And also we set the specific questions for both of the mapping study and the online survey. We have built the protocols in the mapping study and the online survey. (See Appendix A and Appendix B)

- *Developing preliminary synthesis.* We have performed the preliminary synthesis in the three methods. In these methods a variety of tools and techniques were employed, such as textual descriptions for the claims survey (see Chapter 2), groupings & clusters and tabulation for the mapping study (see Chapter 4 and 5) and the online survey (see Chapter 6 and 7). These results of the preliminary synthesis can be used for exploring the relations in the next step.

- *Exploring relations within and between studies.* In this step we explored the relations between the results from the three methods from both the general aspect and from the aspects of the individual patterns.
- *Assessing the robustness of the synthesis.* In this step we perform a validity assessment to assess the mapping study and the online survey, so that we can assess the robustness of the final synthesis.

3.5.4 Participants

In this narrative synthesis the author and his supervisor were involved in the whole process. The author performed the comparison and the data synthesis. Then his supervisor reviewed the synthesis process and evaluated the results.

3.5.5 Paradigm

In the narrative synthesis the results from the mapping study, the online survey, and the claims survey were regarded as the ways of thinking about the design patterns. They were compared and synthesised together to generate a model of thinking about the design patterns, and to find the common issues and differences which influence the application of design patterns.

3.6 Summary

In this chapter we have provided an overview of this research and described all three of the research methods which were employed. This research contains a mapping study, an online survey, a claims survey, and also the evidence synthesis. From them the claims survey has been described in Chapter Two. This chapter mainly focuses on the description of the other three methods.

We adopted the 6Ps framework from (Oates, 2005) to describe the structure of the applied methods. It provided a clear view of the three methods. The mapping study was employed to investigated how extensively the use of software design patterns had been subjected to empirical study, and tried to find the best

effective mechanism for knowledge transfer. The online survey was conducted to investigate the relations between the profiles of users and the design patterns, and also which patterns were considered as useful. The evidence synthesis was used to organise, compare and synthesise the data from the previous studies.

This structure for the methods has provided a solid basis for the following implementations and for interpreting the results. The specific process of implementation of the mapping study and the online survey, and results of these are described in the following chapters.

Chapter 4

PERFORMING THE MAPPING STUDY

4.1 Introduction

The initial research method adopted was to make use of a mapping study. A mapping study (sometimes termed a scoping review) is designed to provide a wide overview of a research area, to establish if research evidence exists on a topic, provide an indication of the quality of the evidence and identify where 'evidence gaps' and 'evidence clusters' may occur (Kitchenham & Charters, 2007). In this chapter, we describe a mapping study we have undertaken. This mapping study is to determine the scale and extent to which empirical studies have been undertaken to determine how effective patterns are as a knowledge transfer mechanism and the forms of evidence for this. Thus it aimed to establish what empirical knowledge about design patterns was available, and how it was organised.

In order to identify specific research 'evidence deserts' and determine appropriate ways of answering them for the research, the following outline of the mapping study was drawn upon according to the guidelines in (Kitchenham & Charters, 2007).

(a) Firstly a protocol was developed based on the experience reported in (Brereton *et al.*, 2007). It is shown in Appendix A.

(b) Research questions for the mapping study were identified as part of the protocol.

(c) In order to collect papers about design patterns the search strategy was defined and applied.

(d) To identify the claims and collect the data from the papers suitable selection criteria were defined (inclusion/exclusion) to identify the paper containing appropriate data.

(e) Then the data synthesis guidelines were applied, subject to the extent of the data found.

(e) The assessment of the study quality (bias/validity) was performed.

4.2 Search Strategy

The search itself encompassed a wide range of computing journals, conferences and the major digital libraries (including IEEE and ACM) access via a range of search engines. For the searching stage, the general scope of the study was identified as being:

- **Population:** Published scientific literature reporting software design pattern studies.
- **Intervention:** Studies involving the use of software design patterns.
- **Outcomes of relevance:** Quantity and type of evidence relating to design patterns.
- **Experimental design:** Any form of empirical study.

Because the *GoF* (Gamma *et al.*, 1995) is regarded as the milestone textbook on design patterns, the start for the search period was chosen as 1995 (publication of GoF). In our searching three rounds of searching were undertaken. The first round electronic searching applied the terms "software + pattern" and covered the period from start of 1995 until the end of 2009. Initially our search was from

1995 to the end of 2007. But to prevent missing some newly published papers about design pattern after two years research, we extended the search to the end of 2009. This electronic search was performed with the following additional keywords:

- **Round 1**: *experience, investigation, experiment, study, experimental, empirical, apply, use, implementation, application, investigate, investigating, experimentation, utilise, utilisation, employ, practice, survey, work, sketch, analyse, analysis, usage, exercise, implement, construct.*

These were used with six search engines/sources: ACM; IEEE Xplore; Google Scholar; CiteSeer; ScienceDirect; and Web of Science. Collectively these addressed the main digital libraries considered to be appropriate to the study. In order to perform a comprehensive searching we checked the different spellings (such as 'utilize' and 'utilise') to treat as synonyms by the search engines. After the electronic online searching two further rounds of searching were performed:

- **Round 2** consisted of a manual search through four journals that were identified as major sources of papers (*IEEE Transactions on SE; Empirical Software Engineering; Journal of Systems & Software; Information & Software Technology*), primarily performed to act as a check on the reliability and stability of the electronic searches, and in particular, upon our choice of search strings.

- **Round 3** consisted of a 'snowball' search, checking the references used in the empirical papers found in the first two rounds, to see if these identified any further papers. This was used to identify any further studies not indexed in any digital libraries that have used other terminology (Davis *et al.*, 2006), such as (Roberts & Johnson, 1996).

During the searching process the study employed a number of selection criteria. These are described in the next part.

4.3 Study Selection Criteria and Procedures

The study selection contains three stages. The first stage was the inclusion/exclusion process on basis of relevance (papers about design patterns). The second stage was the classification process. The third stage was for the empirical papers only, it was a further inclusion/exclusion stage based on the empirical relevance.

For the first stage, immediately following the searching procedure, papers were selected or rejected according to the defined inclusion and exclusion criteria. The study employed the following *inclusion* criteria:

- Papers describing software design patterns (not just empirical papers, as we wanted to be able to categorise the overall patterns literature, although only the empirical papers are actually analysed in this thesis);
- Papers which are published after 1995 (Gamma et al., 1995) were included;
- Where several papers reported the same study, only the most recent was included;
- Where several studies were reported in the same paper, each relevant study was treated as an independent primary study;

and the following *exclusion* criteria:

- Literature that was only available in the form of abstracts or Powerpoint presentations.
- Technical reports or 'submitted' papers.

It should be noted that the choice of search terms meant that the searching process found a wide range of papers other than simply empirical papers, since part of the concern is to determine how well the available empirical studies reflect the issues of concern identified in concept papers and tutorial papers. In order to perform this analysis, therefore it was necessary to identify as wide a range of the patterns literature as possible. At this stage, according to the SLR guidelines, to the searched papers we performed a formal inclusion/exclusion process as follows:

- Exclusion on the basis of title.
- Exclusion after reading the abstract.
- Exclusion after reading the full paper.

In the second stage, the author classified the collected papers according to their purposes. The details about the classification are described in the next section. At the end of this stage all of the collected papers were classified into five categories, and 219 papers were classified as the empirical papers.

On the third stage, we focused upon the papers classified as 'empirical'. Among these empirical papers, a further inclusion/exclusion process was concerned to identify the papers about experience and experiment. This process was performed by the author and his supervisor. The details about this inclusion/exclusion are described in Section 4.5.

4.4 Classifying the Design Pattern Literature

At this stage the goal was to classify the design pattern literature. Firstly, all of the found papers were classified according to their main purpose, namely *Tutorial*, *Support Tool*, *Empirical*, *Construction* and *Index*. As observed above, design patterns have become popular and more and more of the literature describes design patterns. The Patterns Home Page[1] contains pattern definitions and tutorials (Bieman *et al.*, 2003). Besides that it addresses the most frequent approaches which have been used in current pattern research. They are mainly classified as identifying design patterns, writing them up, discussing and teaching them, building support tools, and etc (Beck *et al.*, 1996; Budinsky *et al.*, 1996; Buschmann *et al.*, 1996; Florijn & van Winsen, 1997; Prechelt *et al.*, 2001). The items of the classification are listed as below:

- 'Support Tool' was employed as a pattern theme.

[1] http://hillside.net/

- In the textbook by Gamma et al (1995) the design patterns are classified based on purpose and scope and also the description for how to use a pattern. So for papers about design patterns that describe the internal properties the classification of 'Tutorial' was employed for 'talking about how to use patterns, discussing and teaching them'.

- The external properties of design patterns (Bieman *et al.*, 2003) were noted that the literature describing design patterns can be catalogued (Buschmann *et al.*, 1996; Schmidt *et al.*, 2000). Thus from this aspect it is necessary to know how to index patterns and hence to be able to find patterns. So 'Index' was chosen as one of the classification themes.

- The classification themes mentioned above focus on how to deal with existing design patterns, and as known, design patterns are a set of successful solutions from experts' experience that address recurring problems. So how to write an effective design pattern was another theme in the classification, defined theme as 'Construction'.

- Since the aim of this research was to collect empirical evidence about using design patterns, another classification theme, 'empirical' was used, where this covered different types of empirical study such as Experiment, Case Study, Survey or Experience Report (Jeffery & Votta, 1999; Perry *et al.*, 2000).

Table 4.1 describes the overview of the classification themes and their motivations. Because as mentioned above the aim is to collect empirical evidence for

Pattern Themes	**Motivations**
Tutorial	How to use patterns, discussing and teaching them
Support Tool	Creating and using tools for extracting or using patterns
Empirical	Conducting empirical forms of study
Construction	How to write patterns
Index	How to index patterns and hence to be able to find patterns

Table 4.1: Classification Themes and Motivations about Design Patterns

using design patterns, the empirical papers were further sub-classified. They were classified into four *types* and five *categories*. In terms of the type of study they were classified as *Experiment*, *Case Study*, *Survey* or *Experience Report*. Based on the pattern issue they were categorised as Maintenance, Methodology, Pattern Understanding, Pattern Finding and Other. For these classifications:

- Maintenance was where pattern use was concerned with ease of maintenance;
- Methodology was concerned with how patterns should be used;
- Understanding was concerned with comprehension of patterns;
- Finding was concerned with identifying appropriate patterns.
- A classification of Other was used for the remaining papers.

Table 4.2 displays the four different forms of study for classification of empirical papers, and Table 4.3 shows the classification scheme used for the empirical papers according to the pattern issues involved.

Form of study			
Experiment	Case study	Survey	Experience Report

Table 4.2: Classification of Empirical Papers on Forms of Study

4.5 Study Selection for Empirical Papers

For our initial classification process we took a liberal interpretation of what was meant by 'empirical', including papers that ranged from those that were essentially informal observational 'experience' papers through to those describing controlled experiments. Our motivation here was to ensure that we did not miss any potentially relevant material.

Pattern issues	Motivations
Maintenance	concerns ease of maintenance where patterns are used
Methodology	concerned with how patterns should be used
Pattern Understanding and Evaluation	concerned with comprehension of patterns
Pattern Finding	concerned with identifying appropriate patterns
Other	used for the remaining papers

Table 4.3: Classification of Empirical Papers and Motivations on Pattern Issues

As has been observed in other secondary studies, this process is invariably confounded by the rather casual use of technical terms in many papers. As particular examples:

- Few papers that describe themselves as conducting a case study are actually treating this as an empirical method in the form usually used by the social and 'softer' sciences, such as that documented by Robert Yin (Brereton *et al.*, 2008; Höst & Runeson, 2007; Yin, 2002). Mostly they are observational narratives that could perhaps be more correctly described as 'use cases'.

- The term 'experiment' is used very casually. While in empirical software engineering this is generally taken to refer to a randomised controlled experiment or a quasi-experiment, many authors use the term in a quite cavalier fashion. This is particularly evident in the papers that describe development of tools (such as those used to extract patterns from code) and then describe the application of the tool to a software artifact as being an experiment.

From this point the research initially focused attention on the papers describing experiments and surveys, on the basis that these two forms are least open to experimenter bias. Later, we analysed the empirical papers describing observational experiences. In contrast to the previous two forms of papers, experience papers seem to be more likely to be written by researchers who are advocating the use of patterns.

The 219 papers in the 'empirical' category formed the core set for the formal systematic review process itself. Following the SLR guidelines, to the empirical papers we also performed the formal inclusion/exclusion process as follows:

- Exclusion on the basis of title.
- Exclusion after reading the abstract.
- Exclusion after reading the full paper.

This process was performed by two analysts (the author and his supervisor), and consisted of three phases. Figure 5.5 shows this process. For each phase, we both performed an independent analysis of the candidate papers, and then produced an agreed list. When excluding on the basis of title and abstract we sought to retain any papers that might possibly contain useful experience about the use of patterns, and hence took a conservative approach, retaining a paper if there was any possibility. We also calculated the Kappa score for inter-rater agreement for our initial levels of agreement. For exclusion on title this was 0.44 (moderate agreement), and for exclusion on abstracts it was 0.60 (verging closely on good). The size of the final set is also generally consistent with those found in mapping studies performed on other design-oriented topics. At the end of this process, 13 experiment papers and 7 experience papers were included in our study.

4.6 Summary

This chapter describes the process of the mapping study. We performed a three-round search including an electronic search, a manual search and a snowball search to collect the papers about design patterns. These were then subjected to

an initial filter based on the selection criteria.

Then we classified the collected papers into five categories, namely 'Support Tool', 'Tutorial', 'Index', 'Construction' and 'Empirical'. We then focused on the empirical papers and classified them based on 'Pattern themes' and 'Pattern issues'. After finishing the search and classification, we performed a further inclusion/exclusion step to extract the final included empirical papers. The outcome of the search and classification and the analysis results are presented in Chapter 5.

Chapter 5

OUTCOMES FROM THE MAPPING STUDY

5.1 Introduction

Based on the research method used in the papers, tables showing the distribution were generated. The first round of searching produced 143 empirical papers out of a total of 402. During our Round 2 (manual searching process) 98 relevant journal papers were selected. Within the 98 journal papers there were 42 papers which were classified as the empirical ones. Then snowballing was used to search the papers about design patterns from the references of the empirical papers from the previous rounds. There were 34 papers about design patterns that had been collected and added into the empirical papers database. Since this research was interested in knowing the proportion of papers that had an empirical focus, the other papers (broadly) were classified using the themes of *tutorial*, *support tool*, *empirical*, *construction* and *index* to describe their focuses in Table 4.1 of Chapter 4. Table 5.1 shows the figures obtained from the three rounds. For the initial filtering of these papers a liberal interpretation was performed, so that what was meant by 'empirical' included papers that ranged from those that were essentially observational 'experience' papers through to those describing controlled experiments. The motivation here was to ensure that any potentially relevant material was not missed.

Theme	Interpretation	Round 1	Round 2	Round 3	Total
Tutorial	How to use patterns, discussing and teaching them	107	24	24	155
Support Tool	Creating and using tools for extracting or using patterns	62	25	22	109
Empirical	Conducting empirical forms of study	143	42	34	219
Construction	How to write patterns	37	4	12	53
Index	How to index patterns and hence to be able to find patterns	53	3	19	75
Totals		402	98	111	611

Table 5.1: Distribution of papers across themes

As shown in Table 5.1 the category of empirical papers formed the largest grouping among all of the collected papers. To identify the distribution of the collected empirical publications about design patterns research from 1995 to 2009, the fields of all of the empirical papers such as the type of literature, publication year have been analysed. Through analysis of the data, an overview of the design patterns research over the last decade could be displayed clearly. Table 5.2 provides a summary of the collected empirical papers according to their type. Figure 5.1 displays the distribution of the numbers from Table 5.2 as a chart.

From another point Table 5.3 summaries the numbers of the empirical papers in each year since 1995. Meanwhile for the purpose of intuition on the design

Literature Type	Number
Book Section	9
Conference Paper	117
Journal Paper	92

Table 5.2: Number of Empirical Literature Type

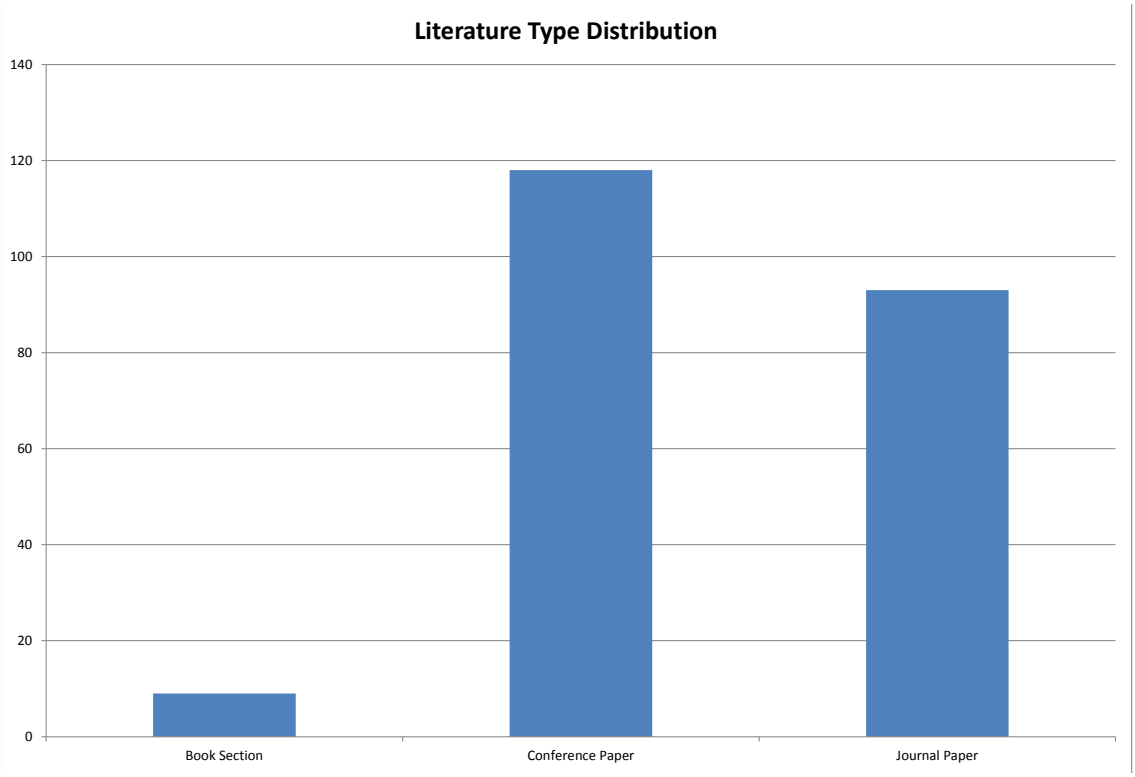

Figure 5.1: Distribution of Empirical Literature Type

patterns research field, Figure 5.2 shows the trend of the empirical literature by publication year.

As can be seen from Table 5.3 and Figure 5.2, during the last decade there was a general rise after 1995, when the milestone textbook GoF was published, in the number of collected papers published. In 1995 there were only 4 papers in our collection. This figure showed that as a new concept in software design, few researchers were concerned with design patterns. After ten years in the year of 2004 the publication of empirical papers about reuse and design patterns reached the peak, there were 27 that year. With the popularity of design patterns, the number of publications about design patterns obviously increased. After 2007 the number of the papers decreased dramatically. In extending our literature search from 2008 to the end of 2009, we used very strict inclusion/exclusion criteria.

Year	Number
1995	4
1996	10
1997	14
1998	12
1999	12
2000	12
2001	15
2002	16
2003	17
2004	27
2005	15
2006	25
2007	26
2008	6
2009	8

Table 5.3: Numbers of empirical papers in each year since 1995

Many papers which were about OO reuse were excluded. But in the years before 2008 we performed a very wide search and considered some areas relevant to the OO design patterns, such as OO reuse and OO development. So compared with the numbers before 2008, the numbers in both 2008 and 2009 were quite small.

Without considering the factor of our inclusion/exlusion criteria, as described above, the concept of design patterns has become more and more popular in research. With the publication of the textbook by the Gang of Four design patterns have been investigated in a slightly increasing level until 2000. After 2000 with the wide implementation of OO design in software development a more rapid increase in papers about design patterns was shown from our search.

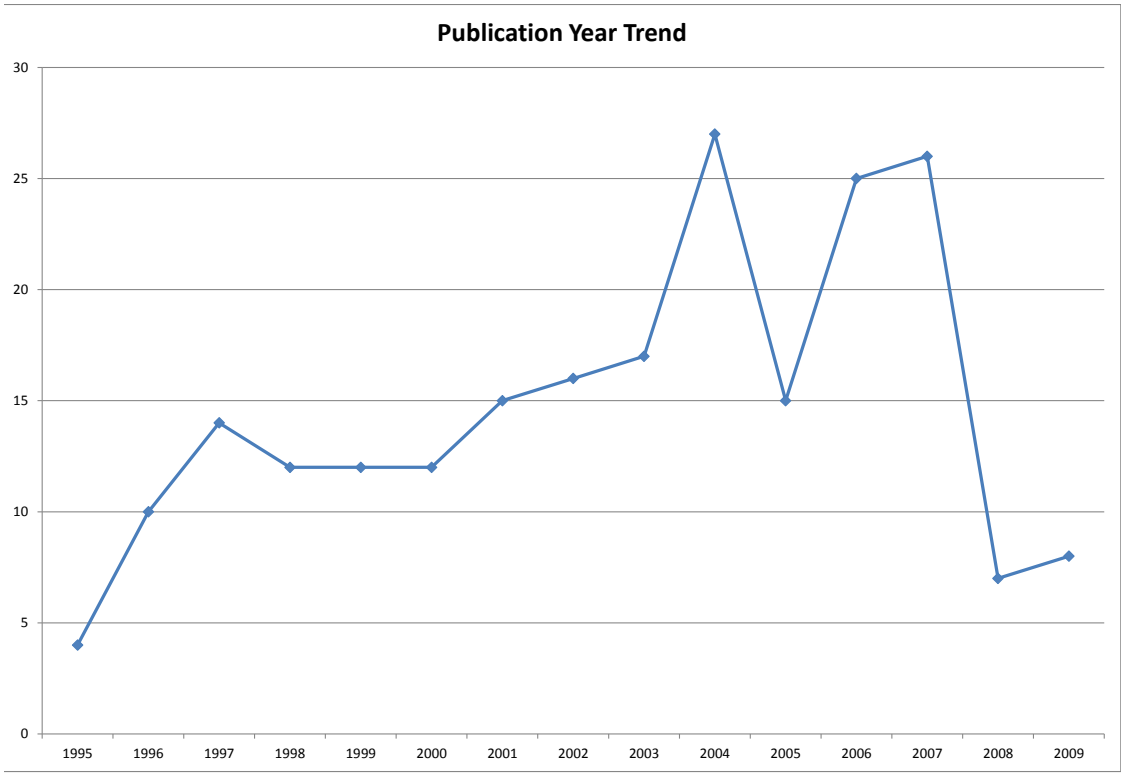

Figure 5.2: The trend of the empirical literature publication by year

5.2 Empirical classification

All the collected papers were classified into the five categories described in Table 5.1. Then the research focused on the category of empirical papers. As can be seen from Table 5.1 there were 219 empirical papers in total out of the whole collection. Chapter 4 described the methods which had been applied in the empirical papers, namely the form of study and the pattern issues.

To sum up the distribution of empirical papers from all rounds in the study, the papers claiming to be a Case Study occupies the biggest part of the distribution: there are 111 Case Study papers in our empirical collection. This is followed by Experiment, Experience, and Survey in order. Table 5.4 and Figure 5.3 list the numbers of papers in each form and display the distribution across the forms of study.

Form of Study	Number
Case study	111
Experience	33
Experiment	62
Survey	13
Total	219

Table 5.4: The number of papers for each form of study

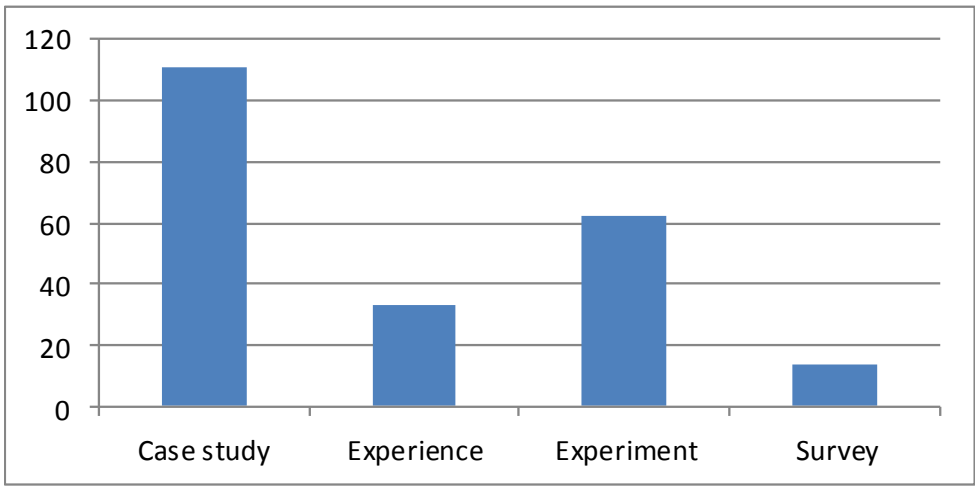

Figure 5.3: The distribution in the form of study

Besides that, in Table 4.3 there is another classification which is based on the aspects of patterns addressed in the empirical papers. This used five categories to classify the problems in the empirical papers. They have been introduced in Chapter 4. As summarised in Table 5.5 and shown in Figure 5.4 the papers about Methodology were the biggest group out of the whole set of empirical papers. There are 67 Methodology papers in total in our collection. And then this is followed by Other, Understanding, Finding and Maintenance in order.

As can be seen from the two kinds of classifications above, the use of a case study

Pattern Issue	Number
Methodology	67
Other	49
Understanding	40
Finding	35
Maintenance	28

Table 5.5: The papers numbers in the pattern issues

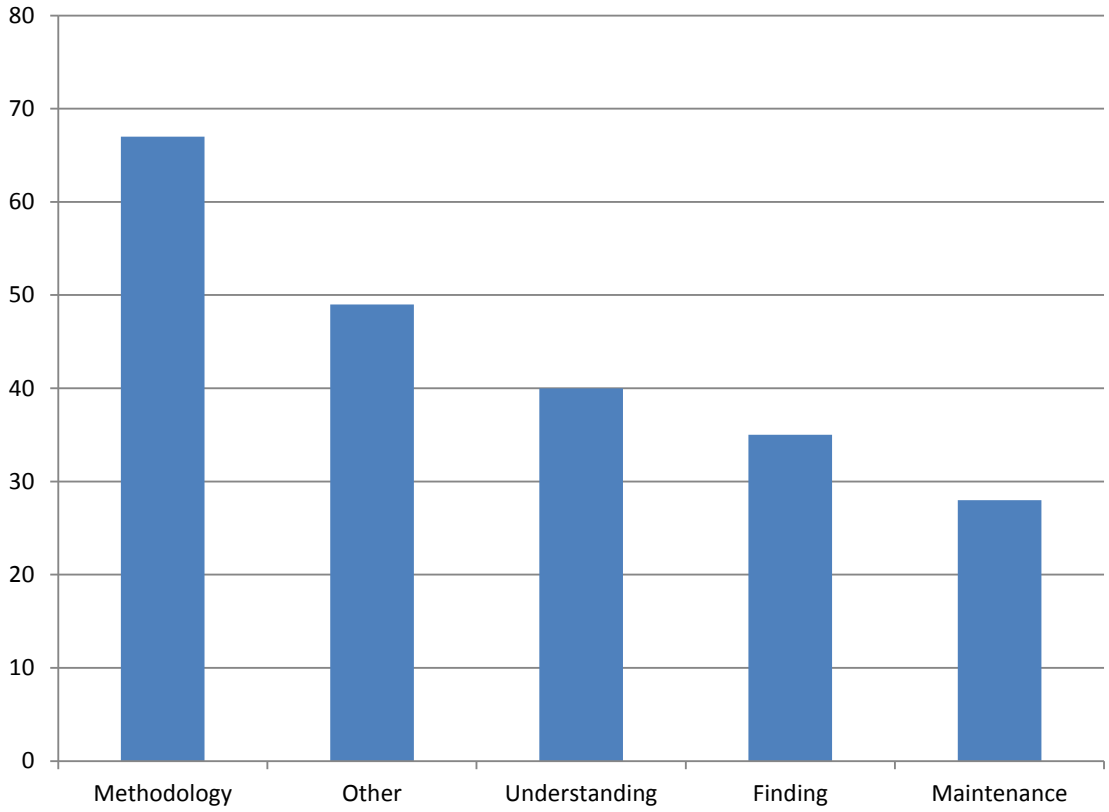

Figure 5.4: The distribution in the pattern issues

is the most frequent empirical method in the design patterns research. And at the same time the most frequent used pattern issue, Methodology means that the most empirical papers focused on how patterns were used during the implementation.

To check the empirical papers we perform the formal inclusion/exclusion process. Firstly we checked the paper titles to exclude some papers. Then we checked the abstracts of the rest of the papers to exclude some papers. Finally we checked the papers by reading the full papers, and decided to include the final papers. Figure 5.5 provides a schematic of this process, indicating the number of papers remaining after each step.

From Figure 5.5, a list of 219 empirical papers was extracted from the collected papers. We then performed the three-stage exclusion/exclusion phase to make final decisions about each paper. In almost all cases it proved difficult to make a definite decision about inclusion/exclusion on the basis of the information in the abstracts, or in some cases, in both the abstract and the introduction section, and for many of the papers we were only able to make a final decision after looking at the complete paper. This process resulted in our keeping only 13 experiment papers and 7 experience papers. The initial filter identified nine candidate survey papers. From the further inspection of these, some papers were not relevant to using design patterns, and some papers were not really survey forms. Therefore this inspection resulted in none of these being considered as relevant. Appendix F and Appendix G show the list of experiment and experience papers respectively.

5.3 Experiment Papers

The 13 experiments involved a range of rigour and experimental form, so they were referred as 'formal studies' (coded as FS1 to FS13) were largely published as Conference or Workshop papers, only three being published in journals. Details of the papers are provided in Appendix F. We should also note that two of these (FS9 and FS11) were conducted as 'replications' of previous studies, although with different types of participant.

This section focuses on what evidence has been extracted from the experiment papers. And then it discusses the details extracted from the experiment papers.

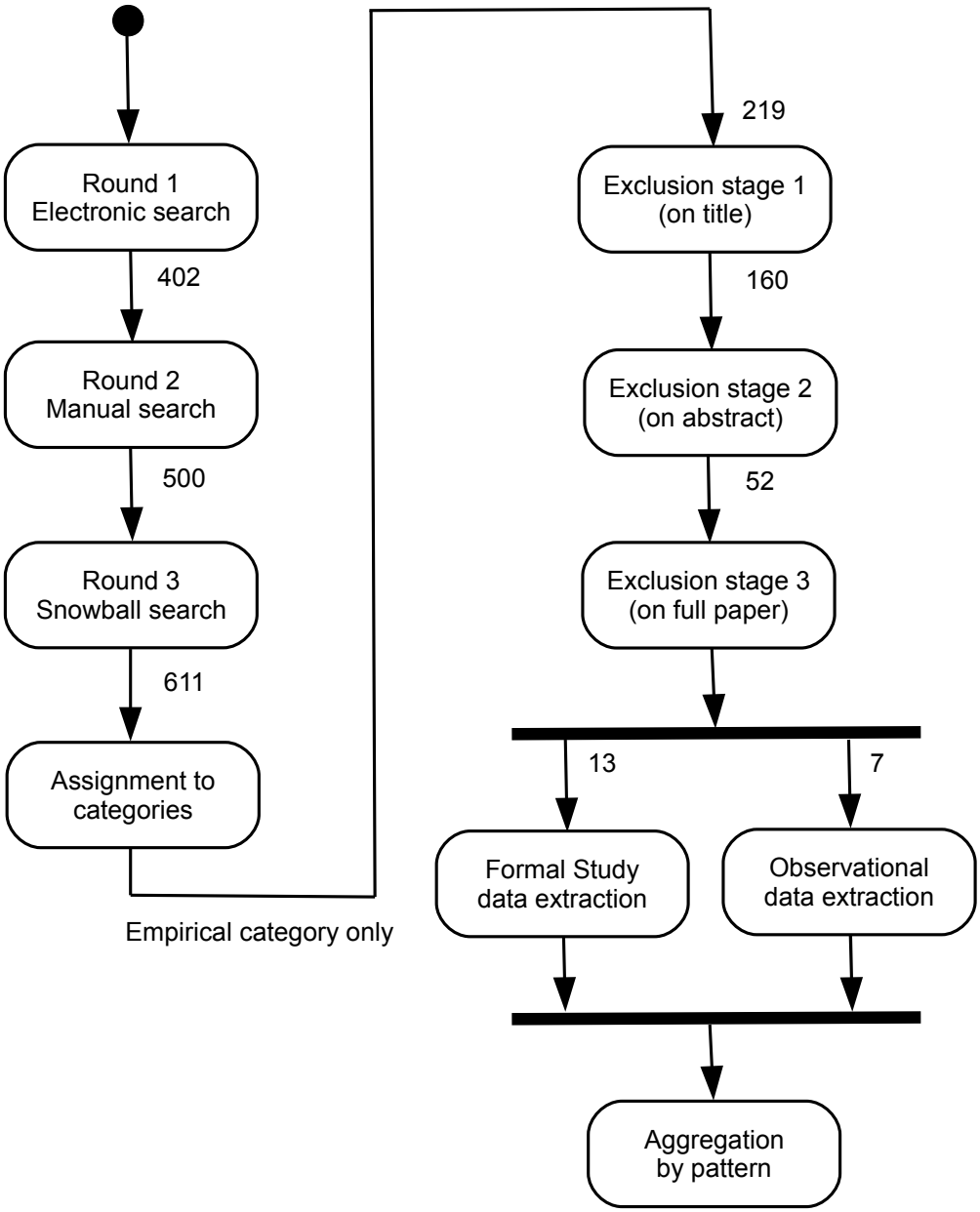

Figure 5.5: Overview of the Mapping Study Process

It began by examining the nature of the evidence provided in the papers that described experiments. To do so, it examines the set of patterns that were studied; the research questions used for the studies; the types of participant and task; and

the extend to which the outcomes of similar or replication studies agreed.

During the more detailed analysis this involved, it became evident that FS6 was a preliminary report on the first part of the study later reported in FS8, with the latter including additional data from a second group who repeated the study in a different venue. To avoid double-counting FS6 was removed from the set, leaving 11 papers. There was also some overlap between FS5, FS7 and FS8, with the first experiment of FS5 being reported in SF8, and the second experiment of FS5 also being reported (more fully) in FS7. So FS5 was also removed from our analysis. FS8 also reported two versions of their experiment, which were performed in two locations, with the different participants, different programming languages, different working modes, and also the different results. So FS8 was treated as separate studies and coded into $FS8_1$ and $FS8_2$. The studies FS9 and FS11 also performed the replications with different profiles. So we kept them for analysis.

5.3.1 Which patterns have been studied?

One study included (FS1) did not study object-oriented patterns and so is not analysed in this subsection (it was included primarily because it studied the use of the pattern concept, but in another domain.) All 11 of the studies on OO patterns used only patterns from (Gamma et al., 1995), and in terms of purpose they used a mix of the 6 creational (C), 13 structural (S) and 22 behavioural (B) patterns that are catalogued there, as well as including both class and object patterns (scope). Table 5.6 summarises the frequency with which particular patterns were employed. Not all of the available patterns from the GoF were used (and several were only used in a single study). The patterns that were not studied were: Builder, Prototype, Flyweight, Proxy, Interpreter, Iterator, Mediator, Momento. Only one study addressed a specific pattern and its properties (FS2). From Table 5.6 it can therefore conclude that only Composite, Observer and Visitor have been studied very extensively.

There was relatively little explanation in the papers as to why particular patterns were chosen. However, for all but one of the papers addressing the use of

Pattern	Purpose	Scope	Studies using	Total
Abstract Factory	C	Class	FS7, FS11	2
Factory Method	C	Object	FS2, FS4, FS13	3
Singleton	C	Object	FS10	1
Adapter	S	Object	FS13	1
Bridge	S	Object	FS10	1
Composite	S	Object	FS4, FS7, $FS8_1$, $FS8_2$, FS9, FS10, FS11	7
Decorator	S	Object	FS7, FS11, FS13	2
Facade	S	Object	FS7	1
Chain of Responsibility	B	Object	FS10	1
Command	B	Object	FS4	1
Observer	B	Object	FS4, FS7, $FS8_1$, $FS8_2$, FS9, FS10, FS11	7
State	B	Object	FS3, FS4	2
Strategy	B	Object	FS3	1
Template Method	B	Class	$FS8_1$, $FS8_2$, FS9	3
Visitor	B	Object	FS7, $FS8_1$, $FS8_2$, FS9, FS10, FS11, FS12	7

Table 5.6: Patterns used in the different experiment studies

OO patterns, the choice was determined by the set of applications employed to provide tasks for the participants, and the choice of these seems to have been based on availability and the need for a tractable size so that participants could understand and change the code in the available time. In some cases, the choice was of course determined by the decision to perform a replication.

The only study that did choose a pattern for a specific reason was FS2, where the purpose was to study the use of the Factory pattern for API design.

5.3.2 Research Questions

We examined the papers to identify the type of research question that they were addressing. These were then used to determine which of the pattern issue categories the paper was addressing (maintenance, methodology, understanding, finding, and other). This is summarised in Table 5.7.

As can be seen, all of the experiments involved an element of 'comprehension', whether or not this was specifically employed for maintenance. Indeed, a very significant characteristic was that all of the experiments on object-oriented patterns were concerned with studying and modifying existing systems. None addressed the development of new software.

5.3.3 Participants & Tasks

Here we report on the type of participant used in each study, and their degree of experience with using patterns. We also examined the tasks assigned to them and the measures used. The summaries of these are presented in Tables 5.8 and 5.9.

Details of the participants were generally reported quite fully and while many were (inevitably) taking advanced undergraduate or graduate courses, there was also quite a large contingent who had extensive programming experience and industry experience. Given the not uncommon view that the use of patterns does require a degree of maturity with object-oriented design and implementation (Sommerville, 2007), this did seem to be recognised in this set of experiments.

A striking feature of Table 5.8 is that most of the OO studies (FS3-FS13) involved modification and that all of them (FS2-FS11) involved an element of coding. Given that design patterns are advocated as being concerned with the more abstract activities of design, such a strong emphasis on coding seems inappropriate-although reflecting something of the difficulty of conducting empirical studies in this area. Indeed, the only study that did not involve an element of coding were FS1, FS12 and FS13.

No.	Research Question(s)	Issue
FS1 (Chung *et al.*, 2004)	How applicable is the pattern concept to ubiquitous computing?	Other
FS2 (Ellis *et al.*, 2007)	Whether factories are detrimental to API usability when compared with using constructors?	Ustand
FS3 (Ng *et al.*, 2006)	Whether a relation exists between the use of patterns and the open-closed principle?	Ustand
FS4 (Ng *et al.*, 2007)	"Given a software system with relevant design patterns deployed and documented, how likely will its maintainer utilize the design patterns to complete an anticipated change?"	Maint
FS7 (Prechelt *et al.*, 2001)	Where a solution using a pattern could be replaced by a simpler one is it still helpful to use the design pattern?	Maint.
$FS8_1$, $FS8_2$ (Prechelt *et al.*, 2002)	"Does it help the maintainer if the design patterns in the program code are documented explicitly (using source code comments) compared to a well-commented program without explicit reference to design patterns?"	Maint.
FS9 (Torchiano, 2002)	(Replication of $FS8_1$.)	Maint.
FS10 (Unger & Tichy, 2000)	If team members have common design pattern knowledge and vocabulary, can they communicate more effectively than without these?	Maint.
FS11 (Vokáč *et al.*, 2004)	(Replication of FS7.)	Maint.
FS12 (Jeanmart *et al.*, 2009)	Whether the presence of Visitor affects developer effort, and whether using different layouts for the UML diagram have any effect?	Maint.
FS13 (Abdul Jalil & Noah, 2007)	To identify the difficulties associated with patterns application by novices.	Ustand

Table 5.7: Research Questions addressed

No.	Details of Participants
FS1	16 pairs of designers, using a mix of professionals (6-8 years of experience) and graduate students
FS2	12 males, aged 18-35 with at least a year's experience of Java, 8 with professional programming experience, 6 with some experience of using the factory pattern that was the topic of the study.
FS3	98 part-time postgraduate students with an average of 5 years working in the computer industry.
FS4	215 undergraduate students taking a Java course.
FS7	29 professional software engineers from one company with average of 2.4 years of C++ experience, and with 15 having some prior knowledge about patterns.
$FS8_1$	74 participants at one site (64 graduate, 10 undergraduate) with an average of 7.5 years Java programming experience.
$FS8_2$	22 undergraduate students with an average of 5 years C++ programming experience.
FS9	28 undergraduate students with limited programming experience.
FS10	15 graduate students with an average of 5.8 years programming experience.
FS11	44 professional software engineers from different companies.
FS12	24 graduate students
FS13	16 final year undergraduate students

Table 5.8: Details of the Participants

No.	Tasks Performed	Measures Used
FS1	Working in pairs. Task 1 to assess whether patterns help with evaluating an existing design. Task 2 to produce a design. Fixed time for both.	Task 1 used participant ratings. Task 2 used ratings from three postgraduate 'judges'.
FS2	5 programming tasks using factory or a constructor.	Completion time.
FS3	Modifying the code of a Java program.	Functional correctness and use of patterns.
FS4	Modifying the code of 3 Java programs.	Functional correctness and use of patterns.
FS7	Modifying two (from four) C++ programs.	Completion time and error count.
$FS8_1$ $FS8_2$	Modify design and then code for two programs.	Completion time and tasks satisfied.
FS9	Modify Java code. (Replicates $FS8_1$.)	Completion time and tasks satisfied.
FS10	Expert/novice pairs performing Maintenance tasks.	Use of protocol analysis (Ericsson & Simon, 1993; Owen *et al.*, 2006) to analyse communication within the pairs.
FS11	Modifying four programs. (Replicates FS7.)	Completion time and degree of functional correctness.
FS12	Interpreting class diagrams.	Focus of eyes on elements of UML diagrams.
FS13	Interpreting class diagrams.	Revised diagrams.

Table 5.9: Tasks and Measures Used

Table 5.10 summarises the details of the different programs used for the purposes of comprehension and/or modification for those studies that used the GoF patterns. For each study we indicate how many programs were employed and indicate their sizes (when known). Where more than one version was provided, we quote the size of the version that used patterns.

No.	Program(s)	Language	Size
FS2	Notepad Email, Elipse Email Thingies, PIUtils, Sockets	Java	*No details provided.*
FS3	Calendar Manager (MCM)	Java	Approx. 1500 LOC
FS4	JHotDraw Calendar Manager (MCM) Hotel Management (HMS)	Java	15815 LOC / 211 classes 1455 LOC / 15 classes 583 LOC / 10 classes
FS7	FS7 Stock Ticker (ST) Boolean Formulas (BO) Communication Channels (CO) Graphics Library (GR)	C++	343 LOC / 7 classes 470 LOC / 11 classes 365 LOC / 6 classes 682 LOC / 13 classes
$FS8_1$	And/Or Tree Phonebook	Java	362 LOC/7 classes 565 LOC/11 classes
$FS8_2$	And/Or Tree Phonebook	C++	498 LOC/6 classes 448 LOC/6 classes
FS9	As $FS8_1$	Java	*Modified, no details*
FS10	CHICO (version control front-end) TIMMIE (time/defect tracking)	Not stated	76 methods/15 classes 102 methods/13 classes
FS11	As FS7	C++	*Slightly different sizes.*
FS12	JHotDraw JRefactory, PADL	n/a	*No Details.*
FS13	Interactive Quiz Environment (IQE)	n/a	*Not given.*

Table 5.10: Programs Used In The Studies

5.3.4 Degree of Agreement between Studies

In order to assess the extent to which the different studies find similar effects this part looked at the following two questions.

- *Do the replication studies demonstrate any degree of consistency in their findings?*

  There are two cases of replicated studies. FS9 uses the material from $FS8_1$ (a study of patterns as documentation) although extending it to use Javadoc tags. Although they are replicated studies, FS9 did not actually focus on the study of patterns. Both studies used student participants and (ignoring the effects of the changes) produced comparable findings. The only difference reported in FS9 was "the average reduction in time is 8%". But FS9 provided opposite results of $FS8_1$. The result of FS8 did not support the hypothesis in $FS8_1$ and provided an explanation for this difference. However FS9 did not provide any explanations for the differences.

  FS11 is a replication of FS7 (comparing the maintainability of designs developed with patterns to those developed without them). Both studies used professional software engineers, but for FS11 these used a real programming environment rather than paper printouts. The conclusions of FS11 emphasised the need to assess patterns on an individual basis and found differences in the results for two patterns (Visitor and Observer). Neither FS9 nor FS11 were close replications (Lindsay & Ehrenberg, 1993), both were differentiated replications, as they made some changes to the form of the experiment. So the lack of any close replications made it difficult to draw any firm conclusions about consistency.

  However both of the replicated studies were performed by participants with a similar background (students for FS9 and FS8, software development professionals for FS11 and FS7). The possible explanations for the differences between them may be the different levels of experience with patterns and the changes in the working conditions.

The variations produced in these two replication studies do suggest that there is a need to perform close replications before differentiated ones. In both cases there was sufficient variation in the experimental conditions to explain the differences in the results, but with no means of determining the actual cause.

- *Where the same pattern is studied in a number of experiments, do these reinforce one another or show different effects?*

  Three patterns have been studied quite widely: Composite, Observer and Visitor. However, since only FS8, FS7 and FS11 made qualitative comments about the effects of specific patterns upon the participant's activities and solutions, even for these, only limited information is available.

  For Composite, for FS8 involved several tasks using the multiple patterns. So it was difficult to separate the actual effects of using the Composite pattern. In FS11, the participants were not familiar with the nature of recursion, so that it caused problems. However the replicated work FS7 did not find any effects.

  For Observer, while FS11 noted no problems, for $FS8_1$ observed that its use led to a significant increase in the time needed to complete a task. FS7 commented that in the one program where it was used, it led participants to create overly-complicated solutions. But the replicated study FS11 did not find any negative effects when compared with observation from the original study FS7.

  For Visitor, FS5 noted that (when used with Composite) its use resulted in less time being taken (for $FS8_1$) or led to fewer errors ($FS8_2$). In contrast, FS7 noted no specific effects, but FS11 considered it complicated, leading to longer development time and poor correctness, to the effect that participants avoided it. FS12 found it still did not reduce the effort of the

participants.

Factors such as task size and background of participants may well have influenced these differences, but they do suggest that the outcomes from individual studies should perhaps not be used in isolation.

5.4 Experience Papers

A list of 22 papers describing experience was extracted from the set of papers classified as experience papers by the author and then checked by the supervisor - looking at the specific patterns reported in each experience paper. This section focused the attention on those papers reporting on the implementation experiences of the specific patterns such as those from GoF (Gamma *et al.*, 1995), rather than on how to use design patterns. Eventually we selected 7 of these.

This section identified the patterns implemented in each experience paper, and then classified the patterns according to the different issues. At this stage the aim was to see if specific patterns are frequently discussed (or patterns from specific libraries) and the forms of experience provided.

The form used for extracting information from the experience papers was subdivided into four main sections as follows:

1. *Q1-4.* Citation details. These are relatively standard.

2. *Q5-11.* Study context. In these questions we sought to identify the characteristics that might influence the way that any outcomes should be interpreted and weighted. We were interested in knowing how independent the authors were (essentially by looking at the references to see if they were authors of patterns or books about patterns); in knowing about the system(s) that provided the source of the experiences; in knowing whether these experiences were first hand or not; the level of abstraction at which the experiences were discussed; and how these were related to the software life-cycle.

3. *Q12-14.* Information provided. Here our interest was in the details of the patterns involved; the conclusions about them; and how these were derived.

4. *Q15.* Decision about inclusion. This simply recorded our final decision.

Table 5.11 identifies the data extracted for each paper. After extraction, the author and the supervisor performed a joint review to make final decisions about each experience paper. The experience papers which just focused on specific patterns and demonstrated the link between patterns and experience lessons were the ones that should be included. This process resulted in rejecting 15 of the papers, keeping only 7. One out of this set is a 'grudging' include because there are no lessons derived about specific patterns (beyond reporting on the choice of patterns used or specific hotspots in the design) (Masuda *et al.*, 1998). But parts of the measures it provides are still valuable, thus this paper could be used to augment any overall conclusion about patterns rather than about the individual patterns.

No.	Pattern items	Motivation
1	Our reference number	The number cited in the empirical papers list.
2	Author	Paper's author
3	Title	Paper's title
4	Where published	The publication source of the paper.
5	Are author(s) pattern developers?	Whether paper author develop the patterns or not.
6	Form of study	proved difficult to code for this group of papers (experience papers are mostly observational, but organised in different ways), and most of the issues involved were essentially captured by the remaining questions. Hence we will not separately analyse the results for Q6.

Continued on next page

Table 5.11 – continued from previous page

No.	Pattern items	Motivation
7	Type(s) of system(s)	few papers reported this information, although it is quite important for the reader who might want to be reassured that the experience provided is relevant to their needs. The first three papers in Appendix G did report the basic purpose of the systems involved (although Wendorff (Wendorff, 2001) gave only a very vague indication). There may well be good commercial reasons for not giving much detail, but that should not prevent enough being given for the reader to be able to determine how it relates to their interests.
8	Size of system(s)	was intended to extract the size of the system that formed the source of these experiences. Only four papers in the set of 22 gave any indication of this, and then all three used totally different measures (KLOC, number of classes, and years of development plus number of developers). Again, this is information that is really needed by the reader in order to assess the relevance and quality of the information-especially as this is one aspect of experience papers that is likely to differ very substantially from controlled laboratory experiments.
9	Experience from own development or of others	addressed the issue of whether the experience was from the authors' own development work or that of others (allowing for the possibility of being both) and extracting this generally proved straightforward. However, there were still papers where this was not made clear to the reader.
10	Design / Coding Forbig & Lammel (2000)	sought to distinguish between experience that was gained from working at a design level of abstraction, or from using the (code-oriented) realisations of patterns. Again, this was usually made evident in the discussion provided in the papers.

Continued on next page

Table 5.11 – continued from previous page

No.	Pattern items	Motivation
11	Development / Maintenance	focused on a rather important distinction, which was whether a paper described experiences gained from development or from maintenance. While the former was by far the most common, maintenance does provide a better retrospective view of the effects that the use of patterns to structure a system can have upon its form and performance.
12	Patterns discussed (with issues raised for them)	The specific patterns being discussed in the paper and details such as Patterns names, Pattern source, Advantages / benefits of pattern and Conclusion.
13	Conclusions	The lessons from the experience.
14	Form of link from experience to conclusions	How to learn lessons from experience.
15	Keep or reject	The paper is included or excluded.

Table 5.11: The data extracted for each pattern in the experience papers

Similarly to the experiment papers, the seven included experience papers (referred as 'observational' and coded as O1 to O7) were largely published as Conference papers, only two being published in journals. As mentioned above, both the author and the supervisor checked the seven experience papers according to Table 5.11 and compared their results. They then concluded by agreeing the result for each item in the seven experience papers after the comparison. Details of the papers and the comparison of coding results between them are provided in Appendix G and the following sections.

For each experience paper included in the mapping study we examined the experiences of specific patterns which are reported in the papers. These patterns details were extracted as Pattern name, Pattern source, Advantages / benefits of pattern and Conclusion. We also examined the lessons from the experiences and

the link between lessons and experiences, namely 'Conclusions' and 'Form of link from experience to conclusions' in Table 5.11.

5.4.1 Lessons learned from the experience papers

This section compares the format of the lessons learned from the experiences of using specific patterns, namely the Summary item in Table 5.16. This is summarised in Table 5.12.

No.	Conclusions
O1 (Schmidt, 1995)	Provided as a summary in the (long) list of lessons
O2 (Yuanhong *et al.*, 1997)	Argues for extensibility as a result of pattern use, but no examples, cautions against indiscriminate use, and emphasises need for care in design.
O3 (Wendorff, 2001)	Conclusions relate in part to misuse clue to each of understanding (cookbook approach), and to misguided attempts to create flexibility. Need objective assessment of the benefit of using patterns.
O4 (Masuda *et al.*, 1998)	Resulting system is larger than no patterns, poorer performance, and greater flexibility. Analysis based on 6 use cases.
O5 (Wydaeghe *et al.*, 1998)	Assessing experience for each pattern in terms of flexibility, modularity, reusability, and understandability.
O6 (Cinnéide & Fagan, 2006)	Assessing the problems can arise in the context of implementing these using Java, and its main limitation is that the industrial experience is distilled into abstract examples rather than cited directly.
O7 (Chatzigeorgiou *et al.*, 2008)	The study notes that "students had difficulties in demonstrating exactly which problems have been solved by each pattern". There is little detailed analysis.

Table 5.12: Lessons learned from experiences of using specific patterns

As can be seen from Table 5.12, most of the experiences about specific patterns are related to the properties of system such as flexibility, reusability and extensibility. Especially in system design, the problem of using patterns appropriately is an issue for concern.

5.4.2 Link between experience and lessons

The previous section described the lessons learned from the experiences. This section focuses attention on how the authors extracted the lessons from the system implementation. Table 5.13 summarises the forms used for linking experience and lessons.

No.	Form of link from experience to conclusions
O1	Uses some examples from the three systems as well as from other sources.
O2	Introduces some experience of using design patterns in the development of JB system.
O3	Based on use of examples to illustrate how inappropriate use may arise.
O4	Uses a decision tree example, does have a systematic approach. No specific lessons about the patterns though.
O5	By using the experiences from developing a customisable diagram editor. Focuses on the different effects noted for behavioural and structural patterns.
O6	Draws upon the industrial experience of one of the authors in order to assess three patterns. main limitation is that the industrial experience is distilled into abstract examples rather than cited directly.
O7	Reports on the experiences of 23 post-graduate students. There is very little in the way of causal links in the reporting.

Table 5.13: Link between experience and lessons

Table 5.13 compares the forms of links from experiences to conclusions in these

seven papers. No matter what kind of lessons they derived from the experiences, all of them used real applications systems as the examples. They all focus on the lessons learned from the experiences towards the question of how design patterns are actually applied and gain benefits from the implementation (Beck *et al.*, 1996).

5.4.3 The details in the studies and forms of implementation

This section focuses more on the details of the studies and the forms of implementation they have used. Table 5.11 listed the pattern items used for checking the experience papers, seven items from No. 5 to No. 11 are used to check what the studies and implementations the papers focus on. Table 5.14 summarises these items and compares them between the four experience papers.

Patterns items	O1	O2	O3	O4	O5	O6	O7
Are author(s) pattern developers?	Yes	No	No	No	No	No	No
Form of study	Develop Reusable Object-Oriented Communication Software, Observation + lessons	Software Maintenance and Reengineering, Observation	System development	Construct decision tree learning systems & identify 'hotspots'	An OMT-editor development	Observations and Lessons from three sample patterns.	Observational study
Type(s) of system(s)	Communication software (including extensive adaption for new systems)	Large commercial project	IDE (JB system)	Decision Tree Learning System	OMT-editor	Commercial software systems	Chosen from a range of topics
Size of system(s)	3 systems (no quoted size)	Large (1000 KLOC)	Sub-system	No quoted size	50KLOC, 173 classes	No quoted size	No quoted size
Experience from own development or of others	Both	Own	Own	Unknown	Own	Own	Others
Design / Coding	Coding	Design	Design	Design	Design	Coding	Design
Development / Maintenance	Development	Maintenance	Development	Development	Development	Development	Development

Table 5.14: Details in the studies and experiences implementation

Table 5.14 concludes that only one author was a pattern developer. Most authors simply applied the patterns from the GoF. To investigate the implemented systems, the authors applied the different systems, and most of them gained their own experiences through the observations made about the systems. In these seven experience papers most focus their attention on system design and system development. Design patterns are used as the means of access to previous successful experiences during the system design and system development processes.

5.4.4 Triangulation with the experiments

After checking all included experience papers, most of the patterns which were extracted from the studies are OO design patterns from the GoF (Gamma *et al.*, 1995). As mentioned in the previous section the paper O4 is a 'grudging' include paper because it does not provide any specific experience about the effectiveness of individual patterns. Therefore it is not included in Table 5.15. Among the other included papers only one study included (O1) was not from the GoF. It studied the Reactor pattern, but this may be viewed as a variation of the Observer pattern which is from the GoF (Gamma *et al.*, 1995). In terms of purpose they used a mix of 3 creational (C), 6 structural (S) and 11 behavioural (B) patterns that are catalogued in Table 5.15 combine with the experiment papers in Table 5.6. Table 5.15 lists the patterns studied in the experiments as well as in the experience papers.

As seen from Table 5.15 only 11 specific patterns were studied in the experience papers. And in these experience papers the Observer pattern could be considered as being studied three times because Reactor may be viewed as a variation of the Observer pattern (Schmidt, 1995). In the experiment papers the Observer, Composite and Visitor patterns are the three that have been studied more extensively. Table 5.15 indicates that the patterns which have the best mix of study types are Observer, Composite and Visitor.

Pattern	Purpose	Scope	Experimental Studies	Experience Studies
Abstract Factory	C	Class	FS7, FS11	O2,O6
Factory Method	C	Object	FS2, FS4, FS13	
Singleton	C	Object	FS10	O6
Adapter	S	Object	FS13	
Bridge	S	Object	FS10	O3, O5, O7
Composite	S	Object	FS4, FS7, $FS8_1$, $FS8_2$, FS9, FS10, FS11	O3
Decorator	S	Object	FS7, FS11, FS13	
Facade	S	Object	FS7	
Chain of Responsibility	B	Object	FS10	O5
Command	B	Object	FS4	O3
Iterator	B	Object		O5
Observer	B	Object	FS4, FS7, $FS8_1$, $FS8_2$, FS9, FS10, FS11	(O1), O3, O5, O7
State	B	Object	FS3, FS4	O2, O7
Strategy	B	Object	FS13	
Template Method	B	Class	$FS8_1$, $FS8_2$, FS9	
Visitor	B	Object	FS7, $FS8_1$, $FS8_2$, FS9, FS10, FS11, FS12	O5
Proxy	S	Object		O3

Table 5.15: Patterns studied in experiments and experience papers

Pattern name	**Pattern source**	**Advantage / benefit**	**Problems / Disadvantages**	**Summary of Experience**
Reactor	Derived by others	Avoid use of threads for handling events from multiple devices	Event Handlers are not preempted; debugging is complicated by flow of control between higher/lower level elements.	Expressed as a set of lessons.
Proxy	GoF	None identified	Intended to aid substitution but adds a level of indirection, and many future changes never materialised.	Overused (unnecessary) and many were removed.
Bridge	GoF	Provides a way to decouple an abstraction and its implementation.	Used where only one implementation of an abstraction - removed these.	Anticipated changes did not materialise.
Command	GoF	Intended to provide a flexible model for command structures.	Was enhanced (unnecessarily) and then change of requirement made unnecessary - but too complex and removed.	Poor design judgement.
State	GoF	Make changing old states and increasing new states locally and easily.	No specific problems	Reduced number of similar statements.
Abstract Factory	GoF	Make it easy to exchange new notation in runtime. Make it easy to extend to support new notation.	Is not as straightforward as it initially.	Cannot provide a complete and robust solution alone

Continued on next page

Table 5.16 – continued from previous page

Pattern name	**Pattern source**	**Advantage / benefit**	**Problems / Disadvantages**	**Summary of Experience**
Singleton	GoF	Applying it lead to very significant performance gains, to resolve global variables issue is one possible solution.	Little known leads to problems.	Avoid using it or make it tolerant multiple instances.
Facade	GoF	Make it more modular and easy to understand.	Increase maintenance costs, longer build times.	None.
Iterator	GoF	Make it easy to add new functionality.	Difficult to understand self-iterator.	None.

Table 5.16: Specific patterns discussed in the experience papers

To investigate each specific pattern we summarise the experiences, classified by Pattern name, Pattern source, Advantages / benefits of pattern, Problems / Disadvantages of patterns and Conclusion catalogues. Table 5.16 displays these classifications for each pattern except the three most extensive studied patterns.

As summarised in Table 5.15, the patterns Composite, Observer and Visitor are the most frequent investigated patterns in both the experiment papers and the experience papers. In the rest of this section we summarise the outcomes from the studies of the three patterns that have been examined most extensively, and analyse the results of this triangulation.

1. *Composite*: Table 5.17 summarises the qualitative assessment for Composite pattern in the experiment papers and the experience papers. From this table the experience papers lack any examples that can be used to compare with the experiment papers. We simply assume that the application of the Composite pattern application should be applied with the knowledge of recursion.

2. *Observer*: Table 5.18 summarises the triangulation result of the Observer pattern between the experiment and the experience papers. There are some conclusions generated by both of them, and also there exist some common points between them.

   O3 and O5 have some agreement on the function of the Observer pattern. Its use could make design more flexible, code more understandable. But the its application is restricted to expert users. O3 also shows some agreement with FS3 on the issue of over-complicated solutions. It increases the complication of the design solution. $FS8_1$, FS11 and O5 agree that the Observer pattern makes design more understandable, but O5 explains that only expert users can meet this positive point. There is also some small reinforcement from one survey (Khomh & Guéhéneuc, 2007), where understandability was seen as being decreased by the use of Observer. To sum up both of the experience papers and the experiment papers about Observer, the successful application of Observer is related to whether users

Paper	Participants	Qualitative Assessment
FS4	215 students	No pattern-specific comments
FS7	29 professionals	No issues identified (only used with other patterns)
$FS8_1$	74 students	No issues identified
$FS8_2$	22 students	No issues identified
FS9	28 students	No issues identified
FS10	15 students	No issues identified
FS11	44 professionals	The reliance of this pattern upon use of recursion was noted ("caused some problems") possibly because of an issue with knowledge of recursion
O2	n/a	No comments other than to indicate satisfactory use

Table 5.17: Summary of Qualitative Assessment for Composite

are familiar with it or not, although it can make a design more flexible and more understandable.

3. *Visitor*: Table 5.19 summarises the qualitative assessment for Visitor pattern in the experiment papers and the experience papers. From this table there are some conflicting views between the experiment papers and the experience papers. $FS8_1$ and $FS8_2$ provide positive results but just with the support of pattern documentation. FS11 is the replicated study of FS7, but its results conflict with FS7. It indicates that the complexity causes lower correctness. FS12 and O5 also support this view.

From the triangulation study between the experience papers and the experiment papers, there are just a few results which can be compared. So synthesis through triangulation could not be performed to a significant degree. That was because the most of the observational studies were reported on non-standard forms and provided only relatively poor data. We have generated a set of preliminary re-

Paper	Participants	Qualitative Assessment
FS4	215 students	No pattern-specific comments
FS7	29 professionals	Its use increased the time needed for understanding and modifying the software when compared to a simpler solution, indicating a risk of overly-complicated structures arising from unnecessary use of this pattern (for the application)
$FS8_1$	74 students	When supported by pattern documentation, led to faster modifications
$FS8_2$	22 students	No specific comments
FS9	28 students	No specific comments (but note earlier observation about the different outcomes for this replication)
FS10	15 students	No specific comments
FS11	44 professionals	No problems reported and generally well understood
O1	n/a	Comments relate to event-driven systems only.
O3	n/a	One overly-complicated solution reported, created by a programmer who wanted to gain experience with using patterns.
O5	n/a	Increases flexibility of a solution and potentially eases reuse. Code becomes more understandable, but only for more expert users who know about the pattern.

Table 5.18: Summary of Qualitative Assessment for Observer

porting standards for experience papers (Budgen & Zhang, 2009), with a hope of encouraging better reporting.

5.5 Summary

This chapter presents the outcomes of the mapping study and analyses the data of the experiment papers and the experience papers respectively. First of all, we described the outcomes of searching and the classifications. Then we analysed the 11 experiment papers and the 7 experience papers. For analysing the experiment

Paper	Participants	Qualitative Assessment
FS7	29 professionals	Task involved an unnecessary Visitor, but effect of this not necessarily viewed as being harmful (data inconclusive)
$FS8_1$	74 students	When supported by pattern documentation, led to faster modifications
$FS8_2$	22 students	When supported by pattern documentation, led to fewer errors
FS9	28 students	No specific comments
FS10	15 students	No specific comments
FS11	44 professionals	Replication contradicts the original study (FS7) and indicates that its complexity led to increased time to perform tasks and lower correctness
FS12	24 students	Does not reduce effort for comprehension tasks, but may reduce effort for modification tasks
O5	n/a	Noted that "for somebody not acquainted with the Visitor pattern, the internal workings of the Visitor may be hard to understand"

Table 5.19: Summary of Qualitative Assessment for Visitor

papers we extracted the information such as 'Research Question' and 'Participants & Task' from these papers. Then we compared the results of extraction for some specific patterns. For analysing the experience papers we created a template to extract the papers' information. Then we concluded the lessons learned from these experience papers. Finally, we performed a comparison between the results of the experiment and experience papers.

The mapping study brought very solid and comprehensive information about the area of using design patterns. As a part of this thesis its result is used to synthesised with other studies in Chapter 8, and the answers for the research questions are presented in Chapter 9.

Chapter 6

PERFORMING THE SURVEY

6.1 Introduction

The previous chapters report upon a mapping study of design patterns. This systematically examined the literature about design patterns and classified the papers formed. The mapping study identified 611 papers about GoF design patterns. Based upon these papers, a major task of the mapping study was to examine the extent of the empirical knowledge available for design patterns. But after performing the mapping study the number of empirical studies that met the inclusion criteria was quite small. There were just 11 papers describing experimental papers and 7 papers describing experience papers which were considered of sufficient quality to be included in the final analysis.

Because of the small number of papers containing empirical evidence about design patterns, and about the usefulness of particular patterns, it was felt that it would be valuable to perform a primary study to try to find out more about which patterns from the 23 in the GoF text were more widely used (and which ones were little used).

The survey is a form of empirical study which is employed for collecting different thoughts and information for a range of issues (Fink, 2002). Nowadays surveys are widely employed in the field of information systems. Based upon to

the description of the advantages and disadvantages of using surveys that is provided by Oates (2005), we here consider the advantages of employing an online survey and how to overcome its disadvantages during the application process. Its advantages include that:

- A survey can cover a wider population. In order to improve the small number of included empirical papers, the survey investigates a more comprehensive population of people who are researching or interested in design patterns.
- Because an online survey provides access to a questionnaire through the internet, when compared with the traditional postal and interview survey, the online survey is more efficient and it can provide sufficient data in a short time for a low overhead.
- A survey can produce a series of numbers to enable a quantitative data analysis to be undertaken. Where appropriate, a quantitative analysis can then be used to show the relationship between the analysed objects.

There are also some disadvantages. But they can be avoided through the appropriate forms of application:

- Surveys focus on breath rather than depth. But as this study aims to investigate the usefulness of all 23 GoF design patterns, rather than the specific technique issues for the specific patterns, this is acceptable. Also, the mapping study has investigated specific patterns by analysing the empirical papers. The comprehensive analysis of the quantitative data from the survey can be used to complement and compare with the data obtained from the mapping study.
- The result from a survey can be directly displayed as quantitative data. The other information which is not related to use of numbers can easily be overlooked. In the design of this survey, this point was avoided by using open-ended questions. The open opinions of participants could be obtained via these questions.

- A survey focuses on sampling a respondent's opinions at a particular time rather than over a period. But this research is just concerned with investigating the overall opinions about all of the 23 GoF design patterns, rather than investigating how those opinions change about these patterns.

Therefore, while recognising the advantages and disadvantages described above, it was felt that performing an online survey could provide a comprehensive overview of the ranking of all 23 GoF design patterns, and improve and complement the analysis from the mapping study. This chapter aims at explaining how the online survey was performed in order to investigate the usefulness of the specific GoF design patterns. It describes the structure and process of the online survey design. Then it describes the process of the pilot testing of the online survey. And finally, it concludes with the process of the survey instrument design and its contributions.

6.2 Survey Design

For this thesis, the online survey was intended to investigate how useful or otherwise the concept of the design pattern had been demonstrated to be in practice. In particular, it wished to identify how useful individual patterns were considered to be (and vice versa) and which ones were actually used to any significant extent.

The purpose of the online survey was therefore to investigate how deeply design patterns influenced software developers' experiences, which design patterns were most frequently applied, the ones they were most reluctant to use, and the reasons for their views. Therefore the title of the survey was defined as "The survey of experience about design patterns".

This survey form consisted of five parts:

- **Introduction**. The survey began by introducing the purpose of the questionnaire. Then it briefly described the structure of the questionnaire, and provided an estimate for the time needed for its completion.

- **Demographic: 'About you'**. In this part the survey began by investigating participants' personal background as related to software design experience. Then it asked about the participants' experiences with using patterns, and their views about those that were familiar to them.
- **Design patterns evaluation (Patterns considered useful)**. This part asked participants to provide a little more information for the three (or fewer) patterns that they considered to be *most* useful.
- **Design patterns evaluation (Patterns considered not useful)**. This part asked the participants to provide a little more information for the three (or fewer) patterns that they considered to be *least* useful.
- **Other**. This final part asked the participants about some miscellaneous issues about their experience and their observations of using other patterns.

The content of the survey questionnaire is presented in Appendix H. The details and motivation for each part in this questionnaire are explained as follows.

6.2.1 Demographic Information

The demographic part aims to investigate the background of the participants. For our questionnaire it consists of seven questions which ask the participants about their names, email addresses, roles in software field, education levels, experiences with Object-Oriented software and design patterns, etc. Besides that, the demographic part also investigates the participants' general familiarly with the 23 GoF design patterns.

- Question 1: Personal Identity.

  In this question we asked the participants to provide their names and email contacts. Initially this questionnaire was designed to be an anonymous one. But in order to prevent duplicate entry and to distinguish the authors (as original sample), we requested both name and email address to distinguish the different participants. Besides that, the website SurveyMonkey recorded the data entry time and the IP address when the participant entered their

data. Therefore under these conditions we ensured the participant to be unique. Beyond this purpose, we also ensured that the data was treated as being anonymous.

- Question 2: Primary Role in Software Development. (**Which of the following best describes your primary role during software development?**)

  This question was used to identify the participants' primary role during they were in the software development process. It applied a closed type question which allowed respondents to choose one option (Brace, 2004). Here four choices were provided to describe the primary role: *Commercial software developer*, *Software researcher*, *Software teacher*, *Student of computing*.

- Question 3: Education Level. (**Highest degree you have earned?**)

  This question recorded the highest level of education a participant had undertaken. This was also designed as a close-ended type question with only one answer. We provided five choices in order: *Associate degree*, *Bachelors degree*, *Masters*, *Ph.D. or equivalent*, *Other*. The choice '*Other*' was employed to provide for those people who did not earn a degree from a university education.

- Question 4: Object-Oriented Experience. (**How many years of experience do you have with Object-Oriented development?**)

  This question was used to investigate the participants' experiences in Object-Oriented software development. Because our research focuses on the use of Object-Oriented design patterns, it is necessary to be clear about the respondents' experiences of Object-Oriented software development. This

question was designed as a close-ended type question with only one option. But different from the previous ones which applied the type of radio buttons, its choices were displayed using a drop-down menu. There are five year-ranges available for answering this question: *Less than 3 years*, *3 to 5 years*, *6 to 10 years*, *11 to 15 years*, *Over 15 years*.

Here there are two points to mention. Firstly, we set 3 years experiences as a basic level. This is because normally 3 years means that the participant has finished at least a basic degree. The participant could then apply Object-Oriented software design. It is helpful to the design patterns application. Secondly, 15 years experience was set as time limit. This is because in this research we focus on the GoF design patterns, and the milestone textbook GoF (Gamma *et al.*, 1995) was published in 1995. Thus it is not possible that a participant with 15 years Object-Oriented Object-Oriented software design experience could be more familiar with the GoF design patterns. Besides that, every five-years was set as a time interval.

- Question 5: Design Patterns Experience. (**How many years of experience do you have with working with design patterns?**)

  Similarly to the previous one question, we applied a one-answer multi-choice question to ask about the participant's experiences on design patterns application. This was also a drop-down menu which contains four choices: *Less than 3 years*, *3 to 5 years*, *6 to 10 years*, *11 to 15 years*. But differently from the previous question, we just provided values to 15 years. That was because when we designed this questionnaire 15 years had passed since publication of the GoF book. Therefore for this question, which was used to investigate design patterns experience, we just set the time scale to cover up to 15 years. Except for this issue, the other choices were the same as for Question 4.

- Question 6: Design Patterns Writing. (**Have you written any patterns (or rewritten any existing patterns)?**)

  The motivation for this question was to provide a context for the participant's attitudes towards to the pattern concept. In our mapping study we found that most researchers and developers just focused on how to apply the current patterns.

- Question 7: General Views on the 23 GoF Design Patterns.

  This question was employed to identify how familiar the participant was with each of the 23 GoF design patterns. The purpose of the question is to elicit the participant's bias towards these 23 design patterns, to use when to comparing the result from the specific evaluation results in the following questions.

  This question applied a matrix & rating type to measure the participants' views on the usefulness of the different patterns. We employed the scale questions which was adopted from (Oates, 2005). The scale questions do not contain 'neutral' options. The participant was asked about how useful or not useful they considered each pattern when they applied it. We provided the option 'Little or No Experience of using this pattern' to address the issue of anyone who is unfamiliar with the pattern. Then we wanted to ensure that respondent expressed a positive/negative view for easier analysis. Then according to how useful the participants consider them, the coding and the interpretation are explained separately:

  - *Very Useful*: Code 5. The participant totally agrees that the pattern is useful during the application;
  - *Useful*: Code 4. Generally they consider that the pattern is useful even if sometimes the pattern is not efficient;

- *Not Very Useful*: Code 3. Generally they consider that the pattern is not useful even if sometimes the pattern is efficient;
- *Not at all Useful*: Code 2. They totally do not agree that the pattern is useful during the application;
- *Little or No Experience of using this pattern*: Code 1. They have no experience on the pattern so that they could not give assess the pattern.

The scale used 5 values for each pattern, from 5 to 1 in order. An illustration of Question 7 is provided in Figure 6.1.

6.2.2 Design Patterns Evaluation

This section explains how the questionnaire was used to investigate the participants' views on the specific patterns. It was divided into two parts. The first one was employed to identify which patterns were considered as the most useful design patterns by the respondent. The second one was employed to identify which ones were considered as not useful.

The structures used for these two parts were the same. Each part requested the participants to identify up to three patterns that they considered to be most useful or not useful. After choosing the pattern the participants were also required to answer five questions that explored their experiences with using this pattern. To provide an understanding of the motivation and structure of these two parts, the example of a useful pattern evaluation is provided below. The questions of the not useful pattern evaluation are essentially the same as those used for the useful pattern evaluation.

- Question 8: Choosing One Useful Pattern.

  The structure of this question is similar to that of Question 7. It employs a one-answer closed type question to investigate which pattern is the participant's favorite according to his/her experiences of using the different patterns. The difference with Question 7 is that it provides one more

* 7. In this section you are asked to provide us with your assessment of the usefulness of each pattern in the book "Design Patterns: Elements of Reusable Object-Oriented Software" by Gamma et al., based upon your experiences with using that pattern.

	Very Useful	Useful	Not Very Useful	Not at all Useful	Little or No Experience of using this pattern
Abstract Factory					
Adapter					
Bridge					
Builder					
Chain of Responsibility					
Command					
Composite					
Decorator					
Facade					
Factory Method					
Flyweight					
Interpreter					
Iterator					
Mediator					
Memento					
Observer					
Prototype					
Proxy					
Singleton					
State					
Strategy					
Template Method					
Visitor					

Figure 6.1: Question 7 from the Online Questionnaire

choice besides the list of the 23 design patterns for the participants. To allow for the possibility that the participant wants to select fewer than three patterns, this question provides a choice "Not Applicable". If this option is selected, the questionnaire moves on to the next part. For example, if a participant does not have any more favorite patterns in Question 8 and

chooses the "Not Applicable" option, the participant is taken to the next part for 'not useful' pattern evaluation. The aim of providing this choice was to help to increase the response rate and improve the quality of the collected data (Iarossi, 2006).

This question is used to investigate the views of the participant about the chosen pattern and the experience of using this pattern in the following questions. Figure 6.2 shows the example of Question 8. The structure of Questions 14, 20, 26, 32, 38 is same as Question 8.

Questions 9-13 aim to investigate the participant's experiences when ap-

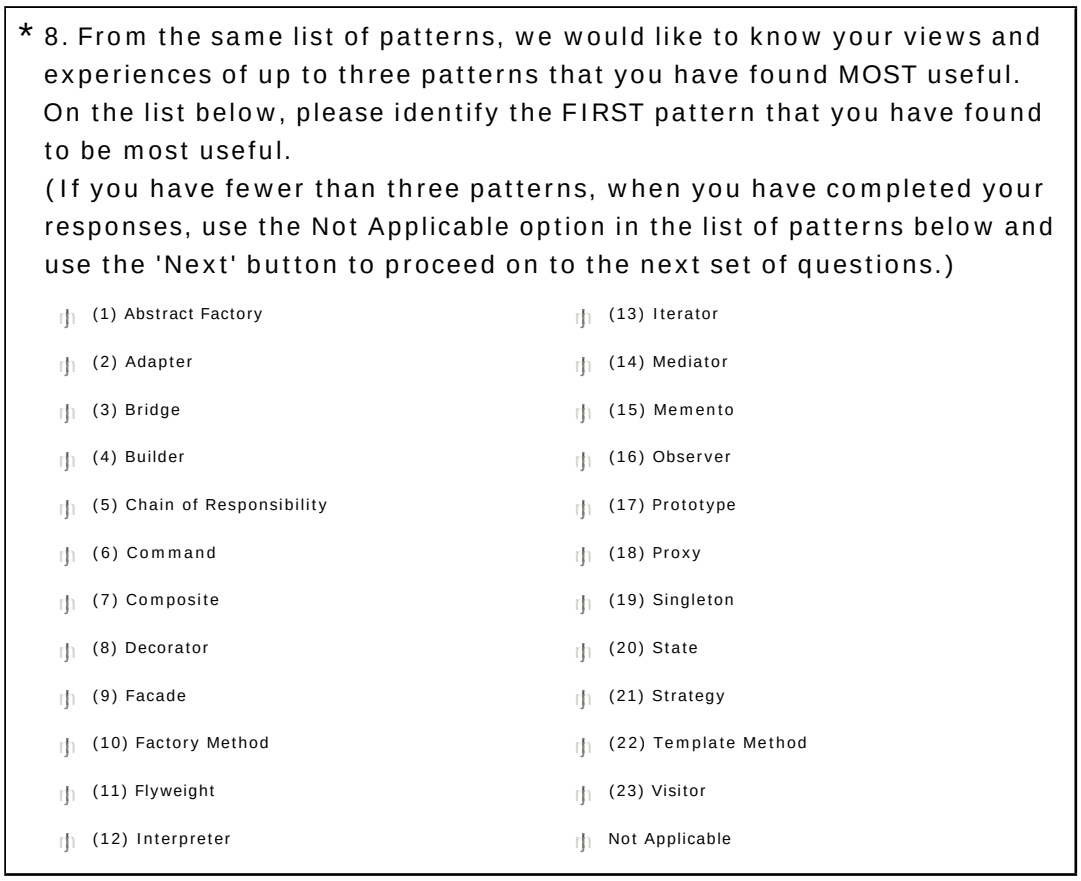

Figure 6.2: Question 8 in the Online Questionnaire

plying the specific pattern.

- Question 9: The Type of Software Created When Applying This Pattern. (**What type(s) of software have you developed or maintained with this pattern? (You can check more than one.)**)

  This question identifies what type(s) of software forms the basis for the participants' experiences with applying design patterns.

  Computer software can be considered to consist of two kinds of program: application software and system software (Naps & Nance, 1995). Application software consists of programs designed to perform different tasks according to different user requirements (Bronson, 2009; Shelly *et al.*, 2009). These requirements are derived from the areas of business, personal, multimedia, communication, etc. Figure 6.3[1] generalised the classification of application software according to its characteristics. Therefore in this question we employed a multi-choice closed type of question which allowed respondents to choose one or more answers. Five choices were provided for this question, based on the software categories above:

  – Productivity/Business Software;
  – Graphic Design and Multimedia;
  – Home/Personal/Education;
  – Distributed systems (such as web-based application);
  – System Software (e.g. Operating system, Middleware).

  Besides that, in case the participants wanted to identify any other type of software experience, this question also provides an option "Other" which is set as an open comment box that can be used to describe the different software type.

[1] Cited from the course slides http://www.sonoma.edu/users/f/farahman/bhcosc/lectures/lec32chap3f04.pdf

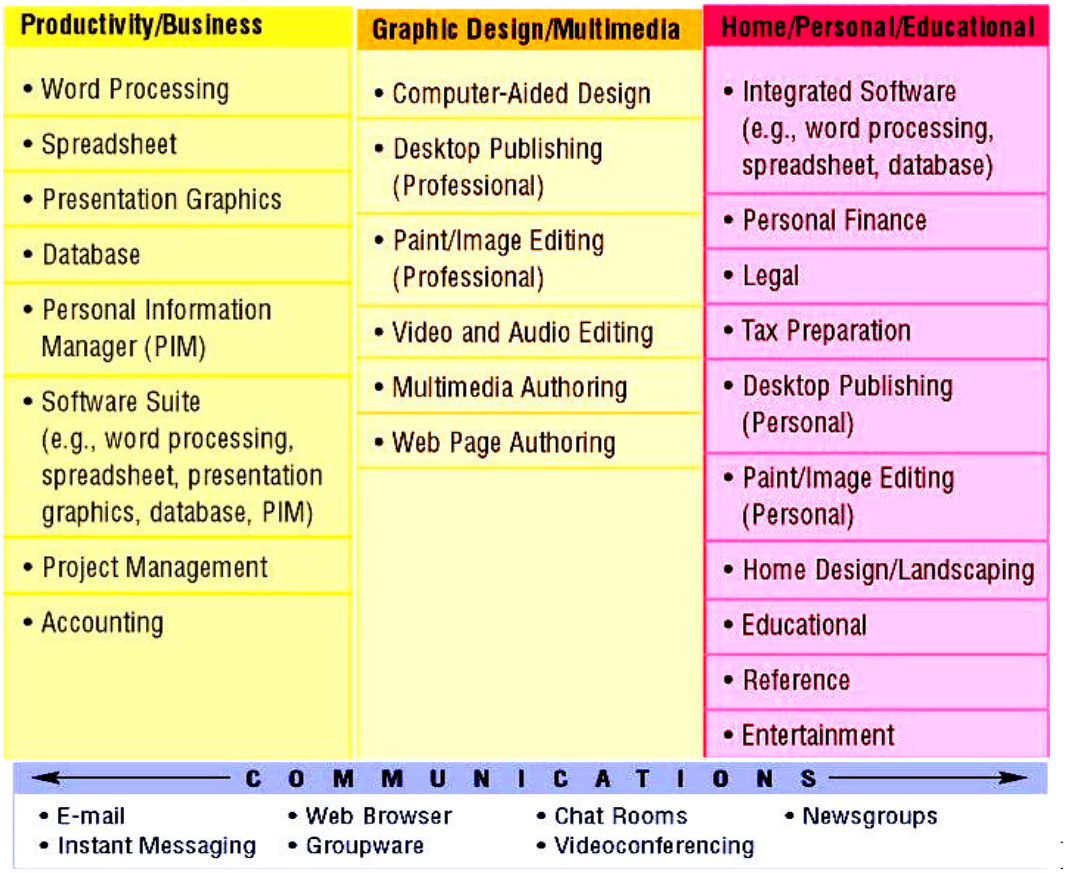

Figure 6.3: Application Software Categories

- Question 10: System Size. (**What was the size of the system?**)

  In order to define the size property of the software providing a participant's experiences, we applied the concept of KLOC to define the system size. KLOC is the abbreviation for thousands of lines of code and is a old metric for software. It can be used to measure the productivity of developers, and indicates the size of system (O'Docherty, 2005). Our survey aims to investigate the effectiveness of reusing design patterns in the participant's experiences. Thus from another side, the size of system reflected by KLOC can potentially measure the productivity of the design pattern. So we employed a one-answer multi-choice question with four scales to reflect

the estimated size of the system: "*up to 100 KLOC*", "*100-250 KLOC*", "*250-500 KLOC*", "*above 500 KLOC*". Here the systems up to 500 KLOC were defined as small and medium-sized systems (Sommerville, 2007). The ones above 500 KLOC were considered to be 'large' systems.

Besides that, to ensure the definition of the system size more precisely we applied another measure of "number of classes" to define the system size. But unlike KLOC we provided an open comment box for the participants, as we expected that they could provide a more accurate value for the number of classes than for KLOC.

- Question 11: Level of Abstraction. (**What was the level of abstraction involved in using this design pattern?**)

  The textbook GoF Gamma *et al.* (1995) concentrated on describing the patterns at a certain level of abstraction. It indicated that design patterns not only focused on object-oriented (OO) design issues, but also had to be based on an object-oriented programming language (Gamma *et al.*, 1995). Therefore in order to investigate which one was employed by the participants when they used design patterns: OO software design, OO code programming, or both of them; we designed a one-answer multi-choice question to ask the participants. This question provides three choices: Design, Code, Both.

- Question 12: Stage in the Life-Cycle of System. (**In what stage(s) in the life-cycle of the system(s) did you work with this design pattern?**)

  Design patterns can help developers choose design alternatives, and improve the maintenance of the existing system (Gamma *et al.*, 1995). So design patterns can be applied in both the stages of development and maintenance in the life-cycle of a system.

This question was designed to identify in what stage the participants' pattern experiences were obtained. In this question we set the life-circle stages *Development* and *Maintenance* as two choices. The participant was expected to choose one answer to reflect the primary basis for their experience.

- Question 13: Open Question for Describing Pattern Experience. (**Please describe the experiences that form your reasons for liking this design pattern, the characteristic of the pattern that you found most useful, and why.**)

  Question 9-12 involved four characteristics of the end software system involved when using design patterns. But it is not possible to cover all of the characteristics of using design patterns, and describe the experience from the participants comprehensively. Therefore we included this open question to ask the participants about their experiences and the reasons for liking the chosen patterns. Also they were expected to identify the characteristics of the patterns that they considered useful and the reasons.

6.3 Pilot Testing

As described in Chapter 3, after developing the online questionnaire, this was reviewed by a small team of two assessors (a 'dry run'). They used the form and also returned the feedbacks by answering a pilot testing questionnaire.

The pilot testing questionnaire contained nine questions and was designed based on the checklist in Fink's handbook (Fink, 2002). It aimed to investigate whether the length of the questionnaire was suitable, whether the structure and the wording was clear and easy to understand, and whether the question choices were exhaustive, etc. Table 6.1 generalises the feedbacks from the two assessors and the improvements we then made towards the feedback from the questionnaire. The two assessors are numbered based on the time of their replies.

As described in Table 6.1, following the feedback from the assessors we made a number of changes to improve presentation and clarity. These improvements mainly concentrated on the fields of questionnaire navigation, question organisation, survey comprehensiveness, and survey instructions - including adding a 'progress report' to keep the respondent informed about how far they had progressed through the form, etc.

Questions	No. 1 Assessor	No. 2 Assessor	Changes
1.Is the length of the survey suitable?	Yes	Yes	None
2.Do the item numbers make the structure of the survey clear to the respondents?		Yes	None
3.Is the layout of the survey clear?	You don't know where you are in the survey. End of page should say how many questions left/pages	Yes	Added a 'progress report' to keep the respondent informed about how far they had progressed through the form.
4.Is the organisation of the questions reasonable?	Yes	I found it difficult to answer the questions about which design patterns were not useful. I can remember thinking about using them but not the details of why I didn't. For example, the size of the developed system. (It is much easier to remember positive experiences!)	Added "Not Applicable" choice for those who cannot decide whether a pattern is useful or not useful pattern in their experiences. (Question 8)
5.Are the questions in the survey easy to understand?	Yes	Yes	

Continued on next page

Table 6.1 – continued from previous page

Questions	No. 1 Assessor	No. 2 Assessor	Changes
6.Are the questions of the survey comprehensive?	Perhaps you should ask if people have used any other patterns - basically ask for name and function. People might not know the number of LoCs in system - could you give them a set of ranges and a don't know option. Or a number of classes option?	Yes	Added an open question to ask whether people used other patterns at the last part of the form; provided a range of KLOC choices and an open box for the number of classes.
7.Are the response choices mutually exclusive?	Yes	Yes	None
8.Are the response choices exhaustive?	Yes	Yes	None
9.Are the instructions for completing each part of the survey clearly written?	Yes	I was initially slightly confused by the questions about the most useful (least useful) design patterns. At first I thought I ought to be doing 1, 2, 3 to identify my favourite three design patterns.	The instructions for each part were rewritten to make them clear.

Table 6.1: Pilot Testing Questionnaire Feedbacks from the Assessors

6.4 Summary

This chapter presents the design of the survey form and how we evaluate the form design. We described the structure of the form, and explained the design details and their motivations for each pattern. The survey form consists of five parts and 47 questions. Each part has its own questions.

Part I is the Introduction part. Part II asks participants to provide their profiles and offer their general assessments for the 23 design patterns. Part III asks participants to provide the experiences of the patterns (up to three) which are considered as useful. Part IV asks participants to provide the experiences of the patterns (up to three) which are considered as not useful. Part VI asks participants some miscellaneous questions and thanks to their contributions.

After finishing the survey form design, we performed a pilot testing. The form was reviewed by a small team of two assessors. Then the forms were sent out with the requests after being modified based on the assessors' comments. The data collection process and the data analysis are presented in Chapter 7.

Chapter 7

RESULTS FROM THE SURVEY

7.1 Introduction

In order to identify which GoF design patterns were considered useful or not useful for software development and maintenance by software experts, we conducted an online survey to focus on this issue. We have described the motivation and the design method applied in the questionnaire form about the GoF (Gamma *et al.*, 1995) design patterns in the previous chapters. In this chapter we investigate the data collection and the analysis operations.

Normally a survey generates both quantitative and qualitative data, as occurred in our survey. In this chapter we employ a statistical approach to analyse the quantitative data collected from the respondents. Statistics forms a kind of science for data collection, data organisation, and data interpretation (Cox *et al.*, 2003). It can be divided into descriptive statistics and inferential statistics (Bernstein & Bernstein, 1999). Descriptive statistics produce tables and charts based on the original data set, and involves collecting, organising, observing, analysing the data through tables and charts to determine the regular pattern of the data. Descriptive statistics focus on the quantitative data (Mann, 2006). Inferential statistics draw conclusions quantitatively for whole population through analysing samplings by using random variation (Bernstein & Bernstein, 1999; Cox *et al.*, 2003).

As described in Chapter 3, the form of our survey involved a mix of exploratory and explanatory analysis. The exploratory analysis as one aspect of data analysis employs a series of methods such as frequency analysis, distribution analysis (Tukey, 1977). These methods are included in the descriptive statistics. For the explanatory analysis inferential statistics are used to validate the hypothesis. In our analysis process we have employed both descriptive and inferential statistics. At the beginning descriptive statistics were applied to analyse the obvious data tendency and patterns. Then for the data sets which did not have obvious patterns between or within groups, we applied inferential statistics to analyse the samplings and to seek to infer the pattern of the whole population. The statistics software tool SPSS was employed to support inferential statistical analysis in our analysis process.

Besides the quantitative data, the qualitative data such as the data collected from the open questions is also analysed in this chapter. The qualitative software tool Atlas.ti was employed to help analyse the qualitative data.

7.2 Data Collection

As described in Figure 3.3, data collection is one element of the last phase in the survey procedure. It consists of the process of administering the survey and the data collection process. This section describes both of these processes and gives a brief analysis of the collected data for each question with tables and charts.

7.2.1 Administering the Survey

As described in Chapter 3, in order to keep the process under control, and to help with organising reminders, we sent our requests out to authors from the list in batches (usually 50), following up each batch two weeks later with a short reminder message to those who had not responded.

With the passage of time, some of the e-mail addresses we had extracted from the papers were no longer valid. In the end we set out some 877 requests, of which

196 were to invalid addresses. Our invitation to participate in the survey also asked recipients to pass it on to others who might be interested, which a number did do. The original author list made it possible to distinguish between those we had asked directly and those who had had the request copied to them. But as promised, all responses were treated anonymously, and the list was used purely for the purpose of assigning responses to a 'response group' at an individual level.

Besides that, we also sent our invitation to three research-oriented mail-lists. The responses from these could not be included in calculations of the response rate, because the size of the mail-lists was unknown. However, because the experts from these mail-lists still can provide valuable responses we included their responses in our analysis.

After sending all the requests and the reminders, we collected 227 responses in total, which were well in excess of expectation. After removing the responses which only answered the demographic questions (Questions 1-6) and administrative questions (Questions 45-47) there were 206 responses left in our database. Because these 206 responses were from three groups, we first of all had to clarify the number of responses in each group. To classify the groups of the respondents, we used three steps:

- Firstly, we extracted the *LinkedIn* mail-list group from the database according to the time of response. The deadline for the authors group was set as 15 June 2010, and we then kept a two-week time gap to aid with partitioning the data sets. After there was no response for two weeks we delivered our questionnaire to the *LinkedIn* groups on 27th June 2010. So we coded the data responses from 28/06/10 as being from the *LinkedIn* group. There are 37 respondents in the *LinkedIn* group.

- After extracting the LinkedIn data there were 169 respondents left. We compared the respondents names with those in the authors list, and extracted those that matched. A total of 128 respondents were extracted from the data, and these were classified as the *authors* group.

- The responses which were left in the data set were classified as being the group of people that the authors passed the survey to. There were 41 responses in this group, termed the *snowball* group.

7.2.2 Profile of the Respondents

Table 7.1 discusses the profile of the final collected responses after completing all of the process described in the previous section.

From this, we sent 877 requests and reminders to the authors of the collected papers. After removing 196 invalid email addresses, there were 681 valid addresses in our database. We received 136 responses from the authors, 53 responses from the people known to authors, and 38 responses from the LinkedIn groups. After excluding the invalid responses we collected 128 responses from the authors, 41 responses from the people known to authors, and 37 responses from the LinkedIn groups finally. Therefore the response rate of the survey originally targeted (authors) is 128/681 or 19%. Because we do not know how many people the authors passed the survey requests to, so we simply added the 53 responses to the sampling frame and calculated the response rate for the wider set as (128+41)/(681+53) or 23%. Because we cannot determine the valid population of the LinkedIn groups, the responses from the LinkedIn groups were not counted in any calculation of the response rate. Whichever value is used as the final response rate, it is well above the usual expected response rate for a survey (10%) as noted in Chapter

	Authors	Snowball of People Known to Authors	LinkedIn Groups
No. of requests	877	unknown	unknown
No. of invalid addresses	196	n/a	n/a
No. received	136	53	38
No. excluded	8	12	1
Final count	128	41	37

Table 7.1: Profile of Responses

3.

Before it is possible to perform any data analysis, it is necessary to be familiar with the profile of the respondents and their distributions. Therefore we analysed the composition of the respondents by making use of the demographic questions in the questionnaire (Q1-Q6).

Analysis of the responses to Question 1 generated Table 7.2, which displays the profile of the respondents based on their primary roles. Table 7.2 shows that the majority of the respondents were software developers and researchers. The numbers for the software teachers and computing students were smaller.

Category	**Authors**	**Snowball**	**LinkedIn Groups**	**Total**	
				(#)	(%)
Developer	20	27	34	81	39.3
Researcher	70	6	3	79	38.4
Teacher	38	1	0	39	18.9
Student	0	7	0	7	3.4
Total	128	41	37	206	100.0

Table 7.2: Profile of respondents: Primary Roles

Using the same categories of respondents, Table 7.3 summaries the distribution of the respondents in terms of their education. Most of the respondents have a degree higher than Bachelors. The respondents who have a PhD degree dominate the majority of all respondents, with 124 respondents forming 60.2% of the entire in the database.

In Questions 4 and 5 of the questionnaire form we respectively obtained the experiences of the participants about Object-Oriented development and design patterns application. We summarised the two profiles for these in Table 7.4 and Table 7.5. Table 7.4 provides a profile of the experience of the different groups of respondents with object-oriented development in general. More than half had

Highest Degree	Authors	Snowball	LinkedIn Groups	Total	
				(#)	(%)
Associate	0	1	0	1	0.5
Bachelor's	1	12	20	33	16.0
Master's	13	18	14	45	21.8
PhD	114	7	3	124	60.2
Other	0	3	0	3	1.5
Total	128	41	37	206	100.0

Table 7.3: Profile of Respondents: Education

over ten years of experience with OO development (54.9%). Table 7.5 provides a similar profile for experience with OO patterns. Again 58 (42.2%) respondents who had 6-10 years experiences of using OO patterns dominated the majority of all respondents. The second group was the group of 11-15 years respondents, 52 respondents (25.2%). From the percentage of each group of the respondents in these two tables, we can see that most of the respondents have at least 3 years experience of OO development and applying patterns. This reflects the expectation that most of the respondents were experts in the areas of OO development and OO design patterns. The probability of the responses being influenced by novices is therefore very small. So in the analysis process of the next step we concentrated on those respondents whose experiences of OO development and using patterns are over three years.

Length of Experience	Authors	Snowball	LinkedIn Groups	Total	
				(#)	(%)
<3 years	1	5	4	10	4.8
3-5 years	2	13	10	25	12.1
6-10 years	35	10	13	58	28.2
11-15 years	34	4	5	43	20.9
>15 years	56	9	5	70	34.0

Table 7.4: Profile of Respondents: Experience with OO Development

Length of Experience	Authors	Snowball	LinkedIn Groups	Total	
				(#)	(%)
<3 years	3	13	14	30	14.6
3-5 years	16	9	12	37	18.0
6-10 years	65	13	9	87	42.2
11-15 years	44	6	2	52	25.2

Table 7.5: Profile of Respondents: Experience with OO Patterns

Table 7.6 the results of Question 6 ("Have you written or rewritten any patterns?"). For this we expected to find that there would be a substantial degree of experience with writing patterns since the original sampling frame was authors of pattern papers. Table 7.6 shows that 122 respondents had pattern authoring experience, the ratio being 59.2%. The distributions in the authors and LinkedIn groups were similar, only for the snowball group was this more balanced.

Pattern Author?	Authors	Snowball	LinkedIn Groups	Total	
				(#)	(%)
Yes	79	20	23	122	59.2
No	49	21	14	84	40.8

Table 7.6: Pattern Authoring Experience

7.3 Data Statistics and Analysis

Normally descriptive statistics can be used to describe and analyse the data based on the tables and charts as above (Mann, 2006) using frequencies, ratio, etc to describe the data patterns (Ross, 1999). But for our samples we found that the descriptive statistics only provided the basic profiles for the data, and we could not analyse them deeply and systematically. Therefore inferential statistics were employed to address this need. This section aims to introduce the statistical forms and analysis methods employed in analysing our survey. These provide the basic and solid concepts for our practical analysis.

7.3.1 Hypothesis Testing

Hypothesis testing is a statistical method used to evaluate a hypothesis and make decisions by using sample data, it is a commonly used method in inferential statistics (Gravetter & Wallnau, 2007; Howell, 1998). Following to the methods for hypothesis testing (Brue, 2005; Howell, 1998; Kirk, 1994), the process applied in our survey data was defined as below:

- Firstly, we defined an appropriate scientific hypothesis about the causes we aimed to investigate. The scientific hypothesis was then used to derive the statistical hypothesis, including the null hypothesis (H_0) and alternative hypothesis (H_A). Normally the null hypothesis represents the case that there is no difference or relation between the samples. If the statistical result shows that there is no significant difference, it means that the null hypothesis is accepted. Otherwise, the null hypothesis is rejected and the alternative hypothesis is accepted if the statistical result shows there exists a significant difference.

- Secondly, we analysed whether the samples satisfied the statistical assumptions or not, and distinguished between the types of data. The condition used for determining satisfaction can be employed to decide what kind of testing is more suitable, parametric testing or non-parametric testing. For example, if we prefer using the ANOVA or MANOVA tests to check the difference of the samples, we have to check the pre-assumptions. There are four assumptions before using the ANOVA or MANOVA tests (Kirk, 1994).

  - *Normality.* The samples have to be selected from a normally distributed population. For our survey the size of the population cannot be determined. But according to the Central Limit Theorem (Rice, 1994), it is approximately normally distributed under the condition of there being sufficient samples.
  - *Randomised.* The samples are selected from the population randomly.
  - *Independent.* The samples are independent and do not influence each other. Normally a survey can satisfy this assumption unless there exists a respondent who is surveyed twice.

– *Homogeneity of Variances*. Generally speaking this is the concept to checking whether two groups of samples are suitable for being compared. If this assumption is not met, the non-parametric test can be an alternative method, or using the "Tamhane's T2" analysing method in "Post Hoc" analysis for multiple compare between the variables.

Besides that, normally the form of the data can be classified as either discrete data, including nominal and ordinal data, or continuous data, including interval and ratio data (Bernstein & Bernstein, 1998). Table 7.7 displays and explains the classification of the data types. For better analysis of the

Continuity	**Category**	**Measurement**	**Example**
Discrete	Nominal	The data is used for classification.	Male, Female
Discrete	Ordinal	The data is in a form of ranking and can be used for calculation.	Very useful, Useful, Not very useful, Not at all useful
Continuous	Interval	The data has a range, and there is an interval between data points.	The age range between 20-25 years old is same as the range 30-35 years old.
Continuous	Ratio	The data includes all properties of other data types. The difference with the interval data is that it has an absolute point.	Kelvin temperature scale is ratio. It has the absolute zero point 273.15K.

Table 7.7: The Classification of the Data

survey we classified the types of the data employed in the survey according to Table 7.7. The clear definition of the data makes it possible to analyse the survey by using SPSS appropriately and determine which statistical test should be employed. For those nominal data elements that have qualita-

tive values we coded them to give them a quantitative property. Table 7.8 displays the definition of the data elements from our survey.

Question	Continuity	Category	Method
Q2	Discrete	Nominal	Quantitative
Q3	Discrete	Ordinal	Quantitative
Q4	Continuous	Interval	Quantitative
Q5	Continuous	Interval	Quantitative
Q6	Discrete	Nominal	Quantitative
Q7	Discrete	Ordinal	Quantitative
Q8,Q14,Q20,Q26,Q32,Q38	Discrete	Nominal	Quantitative
Q9,Q15,Q21,Q27,Q33,Q39	Discrete	Nominal	Quantitative
Q10,Q16,Q22,Q28,Q34,Q40	Continuous	Interval	Quantitative
Q11,Q17,Q23,Q29,Q35,Q41	Discrete	Nominal	Quantitative
Q12,Q18,Q24,Q30,Q36,Q42	Discrete	Nominal	Quantitative
Q13,Q19,Q25,Q31,Q37,Q43 (Open Question)			Qualitative

Table 7.8: The Definition of the Data Used in the Survey

- Thirdly, we chose a suitable statistical test for data analysis. The factor variable and the independent variable were also defined. The data relying on others are defined as the independent variable (Bernstein & Bernstein, 1998). Conversely, the data which are relied by other data are defined as the factor variable.

- Fourthly, we accepted or rejected the null hypothesis based on the analysis results. If the statistical results were not very significant we accepted the null hypothesis (H_0). Otherwise, if the statistical results considered to be statistically significant we rejected the null hypothesis and accepted the alternative hypothesis (H_A).

7.3.2 Data Coding

There are two types of question in our questionnaire. One is the question that provides for one choice. The second is those questions where respondents can make more than one choice. For the questions with only one choice, we employed a Likert Scale (Likert, 1932) and used the meaning of this scale to code these responses. For example, Question 7 was designed to investigate the general views on the 23 GoF design patterns. We asked the participants to evaluate these patterns from the ranked choices: *Very Useful*, *Useful*, *Not Very Useful*, *Not At All Useful*, *Little or No Experience of Using This Pattern*. These choices were coded using 5 values for the options. Table 7.9 presents the example of the coding used for Question 7.

For coding the questions with more than one choice, such as Question 9, we employed another method. Each choice was coded with a value 0 or 1. The value 0 meant the choice was not selected, and conversely, the value 1 meant that the choice was selected. Table 7.10 presents the example of the coding used for Question 9.

Question 7	The Assessment of the Usefulness of Each Pattern				
Variable	Very Useful(VU)	Useful(U)	Not Very Useful(NVU)	Not At All Useful(NU)	Little or No Experience of Using This Pattern (NE)
Value	5	4	3	2	1

Table 7.9: The Code of Question 7

Question 9	The Type(s) of software									
Variable	Productivity or Business Software		Graphic Design and Multimedia		Home or Personal or Education		Distributed systems (such as web-based application)		System Software (e.g. Operating system, Middleware)	
Value	0	1	0	1	0	1	0	1	0	1

Table 7.10: The Code of Question 9

7.3.3 Parametric Test

Parametric test is a statistics method which assumes the data is probability distributed and infer the distribution of the parameters (Geisser & Johnson, 2006). In our survey analysis, we employed the parametric test to infer the distribution of the respondents' profile. From the definition of parametric test, it has to meet the assumptions. If it fails to meet the assumptions, Non-Parametric test can be used for an alternative, and also can apply "Post Hoc" analysis method for multiple compare between the variables. We applied the "Tamhane's T2" analysing method of "Post Hoc" analysis in our parametric test. Parametric test contains a series of methods. In our analysis we applied One-Way ANOVA and t-Test.

- One-Way ANOVA

  One-Way ANOVA is a parametric test to compare means of three or more populations (Bernstein & Bernstein, 1999). The application of One-Way ANOVA in our survey analysis dealt with comparing three groups of respondents' profiles (Hypothesis 1 and Hypothesis 2).

- t-Test

  The difference with One-Way ANOVA is that t-Test compares the means of two populations (Bernstein & Bernstein, 1999). The application of t-Test in our survey analysis dealt with comparing two groups of respondents' profiles (Hypothesis 3 and Hypothesis 4).

7.3.4 Non-Parametric Test

Non-Parametric test is a distribution-free technique, it can be used under the condition of that data do not meet the assumptions of Parametric test (Bernstein & Bernstein, 1999). In our analysis we applied the Kruskal Wallis method and the Chi-Square Test method of Non-Parametric test.

- Kruskal Wallis

Kruskal Wallis is used to test ordinal data. The application of Kruskal Wallis in our survey analysis dealt with comparing three groups of respondents' profiles (Hypothesis 1 and Hypothesis 2).

- Chi-Square Test

  Chi-Square Test is used to test nominal data. It tests that difference of independent sampling groups under a category (Bernstein & Bernstein, 1999). In our survey analysis we employed Chi-Square to test the differences of three respondents' categories on pattern authoring (Hypothesis 5 and Hypothesis 6).

7.4 Data Analysis Process and Results

As described earlier, the questionnaire contained both quantitative data and qualitative data. This section aims to organise, analyse and present results for both of the data forms. In order to analyse the differences between the three respondents groups, the impact of different respondents' profiles on the usefulness of the overall GoF design patterns and each useful & not useful pattern, and the relations between each useful & not useful pattern and the respondents' experience profiles such as software type, system size, abstraction level and life cycle, we analysed the data by using tables and charts. Besides that, we employed hypothesis testing with parametric and non-parametric tests to infer the differences and relation of the variables and factors.

7.4.1 Quantitative Data Analysis

In this section we present the results obtained from analysing the relations between the different variables of each question which were summarised in Table 7.8. A mix of descriptive statistics and inferential statistics were employed in our analysis. The analysing process was based on the structure of the survey. There were three sections in the survey, *demographic*, *patterns considered useful*, and *patterns considered not useful*. In the demographic part we applied the hypothesis method to test the relation between the respondents' profiles and their pattern

assessment. Descriptive statistics were also employed to explore the specific relation between them. The parts of "patterns considered useful" and "patterns not considered useful" focused on the experiences provided for single pattern. For this mainly employed descriptive statistics. Interpretation of the statistical results then provided the report for each section of the survey.

7.4.1.1 Part I: Demographic ('About you')

This part was designed to investigate the respondents' profiles and their summary assessment of the 23 GoF design patterns. Here we brought forward some hypotheses of the relation between the profiles and the pattern assessment, and analysed the results.

1. The Relation between the Respondents' Groups

As shown in Table 7.1, there were three groups of respondents in our database. Comparing the distributions from Table 7.2 to Table 7.5, shows that the profiles for each of these groups differs quite significantly. The *Snowball* group was much more like the *LinkedIn* group than the original *Author* group. Therefore it is necessary to check whether we should analyse the response data from the three respondents' groups separately. The experiences of Object-Oriented development and using design patterns were employed as the variables for checking the hypothesis.

- **Hypothesis 1**: The respondents from the three groups, the original author group, the snowball group, and the LinkedIn group will not be different in terms of their experience of software development.

  H_0: $p > 0.05$. The three groups will not have any significant differences.
  H_A: $p \leqslant 0.05$. The three groups will differ significantly.

  Statistical Tests Used: Kruskal Wallis Test, One-way ANOVA.
  Questions Used: Question 4.

Test Statistics(a,b)

	The three groups would not have significant differences on experience of OO development
Chi-Square	41.624
df	2
Asymp. Sig.	0.000

a. Kruskal Wallis Test

b. Grouping Variable: Group

Table 7.11: Statistics Results for Hypothesis 1

Question 4 inquired about the experience in years with Object-Oriented development. We compared the difference for experience of OO development between the three respondents' groups. Table 7.11 shows that the result has a significant difference ($p < 0.05$). So the null hypothesis is rejected. The three respondents groups do have a significant difference for experience of OO development. Table 7.12 is a complementary result for Hypothesis. It is a result by applying the "Tamhane's T2" analysing method. Although the homogeneity of variances assumption was not met before the One-way ANOVA test and the result was not stable, the result was generated by employing a post test under the condition of the assumption not assumed. So the result also has a referable meaning. From Table 7.12 we find that all p values between the *Snowball* group (2) and the *LinkedIn* group (3) are greater than 0.05. Therefore there is no difference between the Snowball group and the *LinkedIn* group.

- **Hypothesis 2**: The respondents from the three groups, the original author group, the snowball group, and the LinkedIn group will not be different in terms of their experience of using design patterns.

  H_0: $p > 0.05$. The three groups will not have any significant differences.
  H_A: $p \leqslant 0.05$. The three groups will differ significantly.

  Statistical Tests Used: Kruskal Wallis Test, One-way ANOVA.
  Questions Used: Question 5.

Multiple Comparisons

Dependent Variable: How many years of experience do you have with Object-Oriented development?

Tamhane

(I) Group	(J) Group	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Upper Bound	Lower Bound
1	2	1.134(*)	0.226	0.000	0.58	1.69
	3	1.190(*)	0.211	0.000	0.67	1.71
2	1	-1.134(*)	0.226	0.000	-1.69	-0.58
	3	0.057	0.287	0.996	-0.65	0.76
3	1	-1.190(*)	0.211	0.000	-1.71	-0.67
	2	-0.057	0.287	0.996	-0.76	0.65

*. The mean difference is significant at the .05 level.

1 = Author, 2 = Snowball, 3 = LinkedIn

Table 7.12: Multiple Comparisons Result for Hypothesis 1

Test Statistics(a,b)

	The three groups would not have significant differences on experience of using Design Patterns
Chi-Square	50.575
df	2
Asymp. Sig.	0.000

a. Kruskal Wallis Test

b. Grouping Variable: Group

Table 7.13: Statistics Result for Hypothesis 2

Question 5 inquired about the experience year range of using design patterns. We compared the difference on using design patterns between the three respondents' groups. Table 7.13 shows that the result has a significant difference ($p < 0.05$). So the null hypothesis is rejected. The three respondents' groups do have significantly different experience with using design patterns. Table 7.14 also shows the post comparison results by using the "Tamhane's T2" analysing

Multiple Comparisons

Dependent Variable: How many years of experience do you have with working with design patterns?

Tamhane

(I) Group	(J) Group	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
					Upper Bound	Lower Bound
1	2	.879(*)	0.180	0.000	0.43	1.32
	3	1.199(*)	0.166	0.000	0.79	1.61
2	1	-.879(*)	0.180	0.000	-1.32	-0.43
	3	0.320	0.227	0.414	-0.23	0.87
3	1	-1.199(*)	0.166	0.000	-1.61	-0.79
	2	-0.320	0.227	0.414	-0.87	0.23

*. The mean difference is significant at the .05 level.

1 = Author, 2 = Snowball, 3 = LinkedIn

Table 7.14: Multiple Comparisons Result for Hypothesis 2

method between the three respondents' groups under the condition of One-Way ANOVA test do not meet the assumptions. From Table 7.14 we find there is no difference between the *Snowball* group and the *LinkedIn* group. Therefore, we could compare the author group and the LinkedIn group to check the consistency.

Therefore combining the results of Hypothesis 1 and Hypothesis 2, the results show that the profiles for each of these groups differs significantly, especially between the author group and the others. For more precise statistics we analysed the author group separately in the following hypothesis and compared between the Snowball group and the LinkedIn group.

- **Hypothesis 3**: The respondents from the two groups, the *Snowball* group and the *LinkedIn* group, will show no difference for the experience of OO development.

  H_0: $p > 0.05$. The two groups will not have significant differences.
  H_A: $p \leqslant 0.05$. The two groups will have significant differences.

Statistical Test Used: t Test.
Questions Used: Question 4.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Upper	Lower
How many years of experience do you have with Object-Oriented development?	Equal variances assumed	1.218	0.273	0.196	76	0.845	0.057	0.289	-0.520	0.633
	Equal variances not assumed			0.197	75.953	0.844	0.057	0.287	-0.516	0.629

Table 7.15: Statistics Result for Hypothesis 3

We compared the difference on experience of OO development between the two respondents' groups, the Author group and the LinkedIn group. Table 7.15 shows that the result has no difference ($p > 0.05$). So the null hypothesis is accepted. The two respondents groups have no a significant different experience on experience of OO development.

- **Hypothesis 4**: The respondents from the Snowball group and the LinkedIn group will have no difference on the experience of using design patterns.

  H_0: $p > 0.05$. The two groups will not have significant differences.
  H_A: $p \leqslant 0.05$. The two groups will have significant differences.

  Statistical Test Used: t Test.
  Questions Used: Question 5.

Independent Samples Test

		Levene's Test for Equality of Variances		t-test for Equality of Means						
		F	Sig.	t	df	Sig. (2-tailed)	Mean Difference	Std. Error Difference	95% Confidence Interval of the Difference	
									Upper	Lower
How many years of experience do you have with working with design patterns?	Equal variances assumed	3.233	0.076	1.397	76	0.167	0.320	0.229	-0.136	0.776
	Equal variances not assumed			1.407	75.839	0.163	0.320	0.227	-0.133	0.772

Table 7.16: Statistics Result for Hypothesis 4

We compared the difference on using design patterns between the two respondents' groups, the Author group and the LinkedIn group. Table 7.16 shows that the result has no difference ($p > 0.05$). So the null hypothesis is accepted. The two respondents' groups have not a significant different experience on using design patterns.

Combining the results of Hypothesis 3 and Hypothesis 4, the results show that the profiles for each of these groups have no a significant difference, and can be considered as being drawn from the same population. Therefore both of these two groups were merged into a single group. We analyse the data from the author group and the merged group separately in the following hypothesis.

2. The Relation between Patterns Experience and the Usefulness Profile

In order to investigate the relations between the experiences of using design patterns, we compared the profile of choices in Question 7 against the different design patterns experience year range in Question 5. First of all, the numbers

for each assessment choice, such as very useful (VU); useful (U); not very useful (NVU); not at all useful (NU); Little or no experience of using this pattern (LE), were counted under the category of the different design patterns experience. Because the number of respondents in the different experience groups were not the same, it is not possible to compare the count values directly. So we counted the percentages of each assessment in each experience group. Then the assessments of very useful (VU) and useful (U) were aggregated into the positive assessment, and not very useful (NVU); not at all useful (NU) were aggregated into the negative assessment. Table 7.17 shows the numbers and the percentages of the different assessment in the author group. Table 7.18 displays the results after the aggregation.

In the same way as for the author group, we performed the same calculations

Pattern Experience	<3 years	3-5 years	6-10 years	11-15 years
VU	10 14.49%	98 26.63%	512 34.25%	422 41.70%
U	28 40.58%	136 36.96%	499 33.38%	359 35.47%
NVU	1 1.45%	38 10.33%	192 12.84%	87 8.60%
NU	0 0.00%	1 0.27%	42 2.81%	24 2.37%
LE	30 43.48%	95 25.82%	250 16.72%	120 11.86%
Total	100.00%	100.00%	100.00%	100.00%

Table 7.17: The Assessment Totals and Percentage of the Different Design Patterns Experience in the Author Group

and comparisons between the different experiences of using design patterns in the merged group. Table 7.19 shows the numbers and the percentages of the different assessment in the author group. Table 7.20 displays the results after the aggregation.

Pattern Experience	<3 years	3-5 years	6-10 years	11-15 years
Positive	55.07%	63.59%	67.63%	77.17%
Negative	1.45%	10.60%	15.65%	10.97%
LE	43.48%	25.82%	16.72%	11.86%
Total	100.00%	100.00%	100.00%	100.00%

Table 7.18: The Aggregated Assessment Percentage of the Different Design Patterns Experience in the Author Group

We compared the assessment percentages in both of the author group and the

Pattern Experience	<3 years	3-5 years	6-10 years	11-15 years
VU	124	137	177	78
	19.97%	28.36%	34.98%	42.39%
U	148	137	181	66
	23.83%	28.36%	35.77%	35.87%
NVU	68	44	50	25
	10.95%	9.11%	9.88%	13.59%
NU	10	21	6	4
	1.61%	4.35%	1.19%	2.17%
LE	271	144	92	11
	43.64%	29.81%	18.18%	5.98%
Total	100.00%	100.00%	100.00%	100.00%

Table 7.19: The Assessment Number and Percentage of the Different Design Patterns Experience in the Merged Group

merged group. Figures 7.1 and 7.2 compared the assessment percentages in each year range for patterns experience. From these two figures it is very obvious that the positive assessment percentage of using patterns increases with greater experience with design patterns, and the 'less or no experience' assessment percentage of using patterns decreases with design patterns' experience. In the negative assessment there is no significant difference between the different pattern experiences. To summarise, the degree of experience with using design patterns influences how

Pattern Experience	<3 years	3-5 years	6-10 years	11-15 years
Positive	43.80%	56.73%	70.75%	78.26%
Negative	12.56%	13.46%	11.07%	15.76%
LE	43.64%	29.81%	18.18%	5.98%
Total	100.00%	100.00%	100.00%	100.00%

Table 7.20: The Aggregated Assessment Percentage of the Different Design Patterns Experience in the Merged Group

people apply the design patterns. Greater experience means that more patterns are considered useful. Note that we did not consider those respondents whose experience with using design patterns was less than 3 years in the following analysis.

3. The Relation between the Primary Roles and the Pattern Usefulness

We also looked at how the 'votes' were used by the respondents according to their different roles (researchers, teachers and developers). To do this, we compared the assessment percentage in each role. The method is same as used for the comparison of the different design patterns experiences in the previous section.

Figures 7.3 and 7.4 show that there are no significant differences on the assessment of design patterns between the different roles. Within the four primary roles just that of the student is slightly different from the other roles. The percentage of students in Figure 7.3 is 0%. That is there are not any student respondents in the Author group. Figure 7.4 reflects the percentage of design patterns under 'little or no experience' is higher than the other roles, such as developer, researcher and teacher. With the results of the previous section for the pattern experiences, it is clear that the students have less experience than the other roles. Therefore, from the two figures opinions about how to use and assess the design patterns is not influenced by people's primary roles.

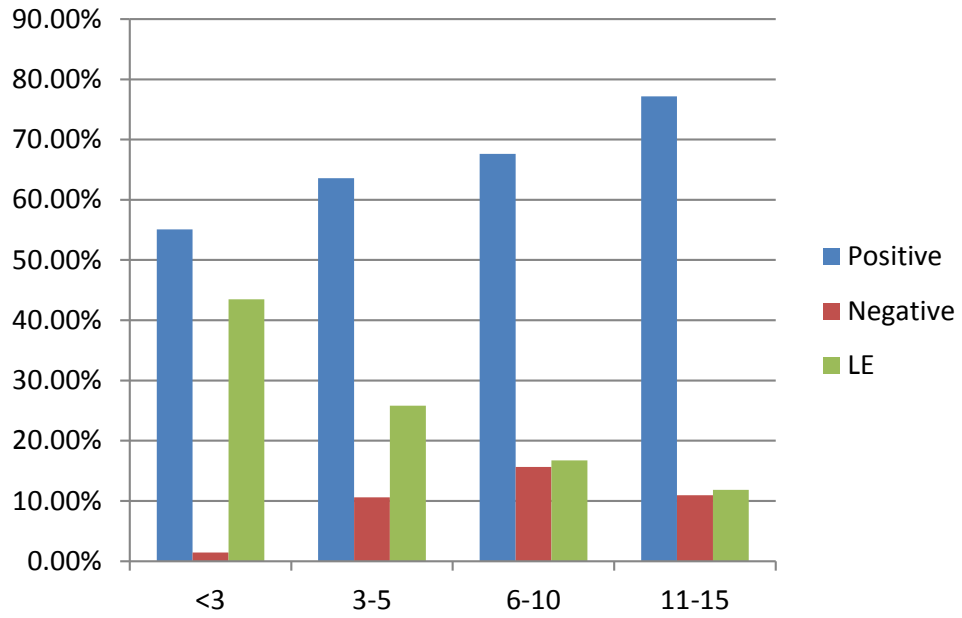

Figure 7.1: The Comparison of Using Patterns Experiences in the Author Group

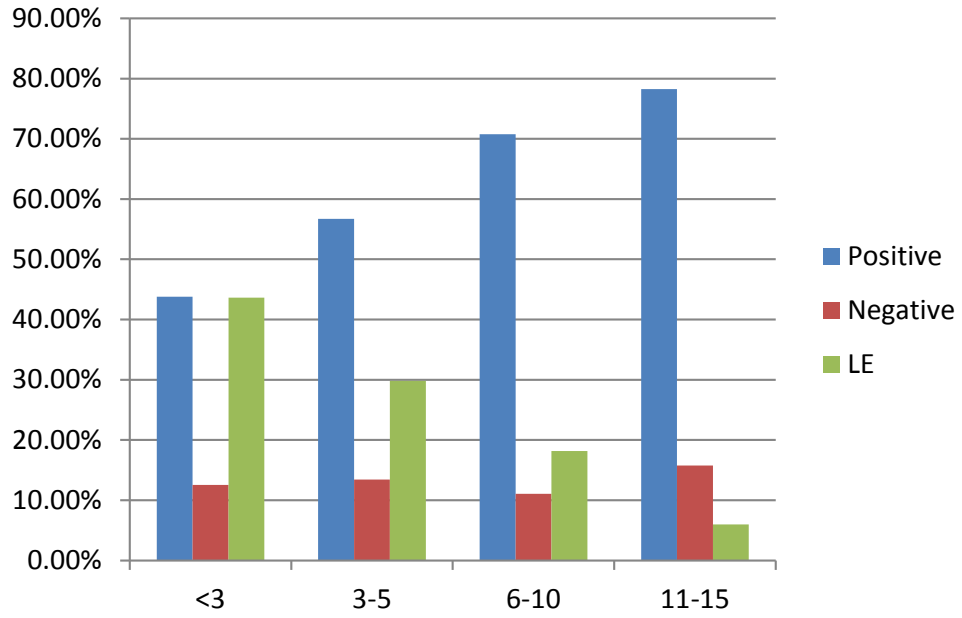

Figure 7.2: The Comparison of Using Patterns Experiences in the Merged Group

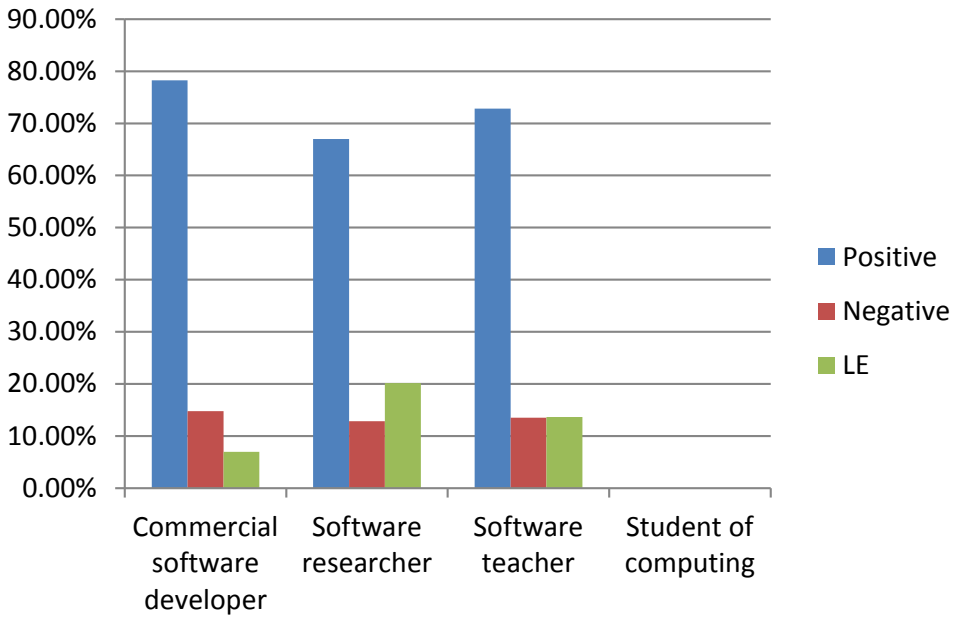

Figure 7.3: The Comparison of Different Roles in the Author Group

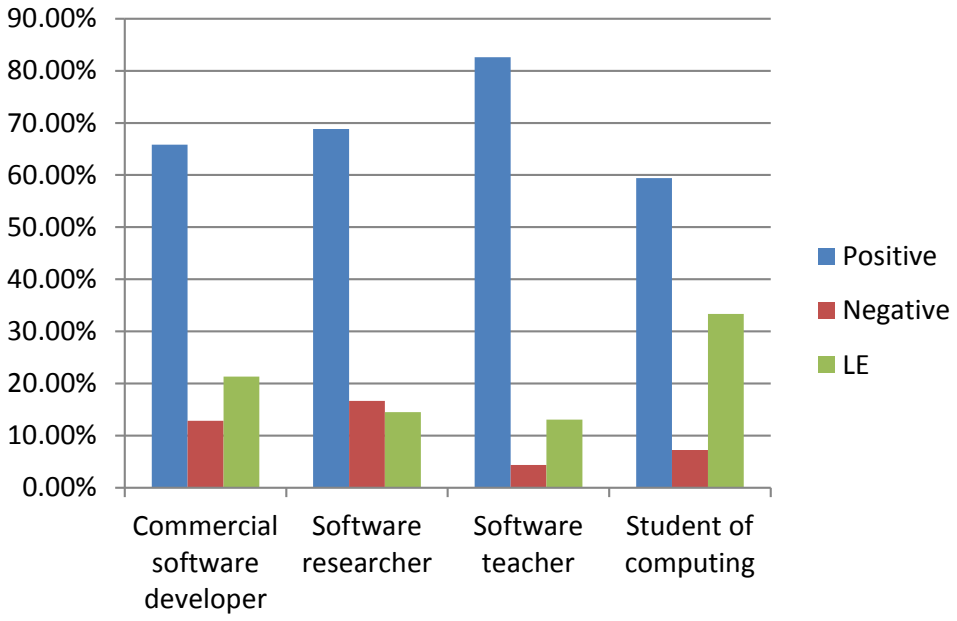

Figure 7.4: The Comparison of Different Roles in the Merged Group

4. The Relation between the Pattern Experience and Pattern Writing

In the mapping study we performed previously, most of the papers focused on the 23 GoF design patterns. But still there were also some papers which focused on writing patterns. Most of these patterns were created based on the 23 GoF design patterns. This motivated us to investigate the relation of the pattern writing and people's experience. Question 6 was designed to investigate the relation between pattern experience the pattern writing experience. Here two hypotheses were employed.

- **Hypothesis 5**: In the author respondent group there will be no difference between the three respondent categories (3 to 5 years, 6 to 10 years, 11 to 15 years) for different levels of design patterns experience and experience with pattern authoring.

  H_0: p $>$ 0.05. The three categories will not have any significant differences.
  H_A: p $\leqslant$ 0.05. The three categories will have any significant differences.

  Statistical Test Used: Chi-Square Test.
  Questions Used: Question 5, Question 6.

Question 5 asked respondents about their experience in years of using design patterns, while Question 6 inquired whether they had experience of authoring patterns . We computed for the significant level over the different pattern experience categories on pattern authoring. Table 7.21 shows that the result p=0.006<0.05, which means the three categories do show significant differences for pattern authoring. So the null hypothesis is rejected. The three categories of using patterns have a significant difference on pattern authoring.

- **Hypothesis 6**: In the merged respondent group there will be no difference between the three respondent category (3 to 5 years, 6 to 10 years, 11 to 15

Test Statistics

	How many years of experience do you have with working with design patterns?	Have you written any patterns (or rewritten any existing patterns)?
Chi-Square(a,b)	29.008	7.688
df	2	1
Asymp. Sig.	0.000	0.006

a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 41.7.
b. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 62.5.

Table 7.21: Statistics Results for Hypothesis 5

years) for different design levels of patterns experience and experience with pattern authoring.

H_0: p $>$ 0.05. The three categories will not have any significant differences.
H_A: p $\leqslant$ 0.05. The three categories will have significant differences.

Statistical Test Used: Chi-Square Test.
Questions Used: Question 5, Question 6.

As with Hypothesis 5, we computed the significance level over the different pattern experience categories on pattern authoring in the merged group. But Table 7.22 shows that the result p=0.327>0.05, which means that the three categories do not have significant differences for pattern authoring. So the null hypothesis is accepted. The three categories of using patterns show no significant difference for pattern authoring.

Since the results of Hypothesis 5 and Hypothesis 6 generated different results.

Test Statistics

	How many years of experience do you have with working with design patterns?	Have you written any patterns (or rewritten any existing patterns)?
Chi-Square(a,b)	7.176	0.961
df	2	1
Asymp. Sig.	0.028	0.327

a. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 17.0.
b. 0 cells (.0%) have expected frequencies less than 5. The minimum expected cell frequency is 25.5.

Table 7.22: Statistics Results for Hypothesis 5

In order to explore the relation of the pattern experience and the pattern authoring, we compared the authoring percentage in the different pattern experiences. Figure 7.5 and Figure 7.6 show the percentage comparison between the different design patterns experience categories in the author group and the merged group. Figure 7.5 obviously shows that in the Author group the percentage of the respondents with pattern authoring experience, increases with years of pattern experience. In contrast the percentage of the respondents, which do not have pattern authoring experience, decreases with the growth of the pattern experience years. For the merged group, Figure 7.6 does not provide any very clear trend regarding pattern authoring for the different experience categories. But they still appear to have a general trend similar to that shown in Figure 7.5.

To sum up all the material above, the pattern authoring is influenced by the experience of using design patterns. More experiences users write their own patterns, seems more likely. It also verifies the importance of experience to the pattern usefulness from another aspect.

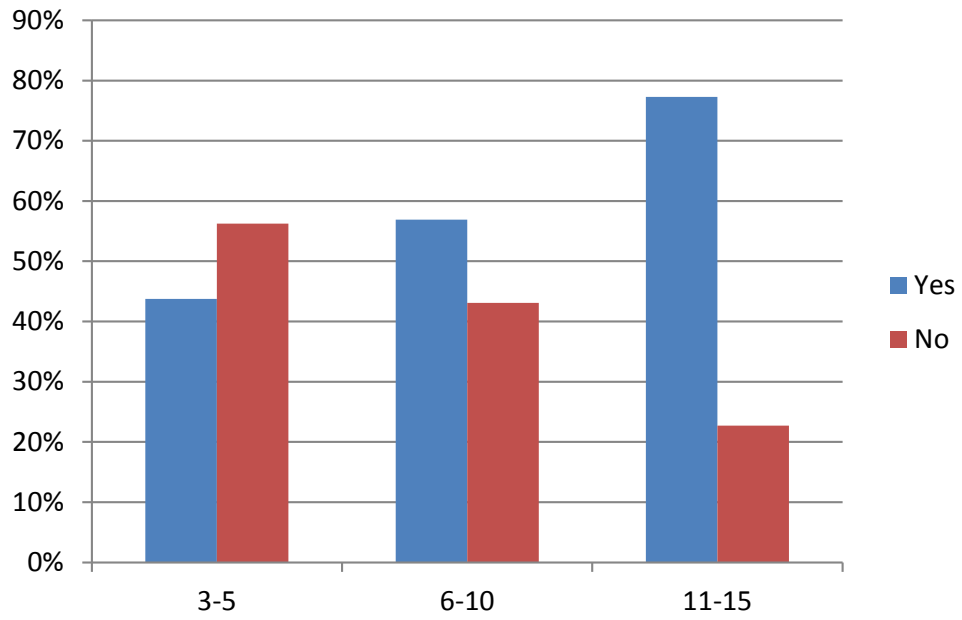

Figure 7.5: The Comparison of Pattern Writing in Author Group

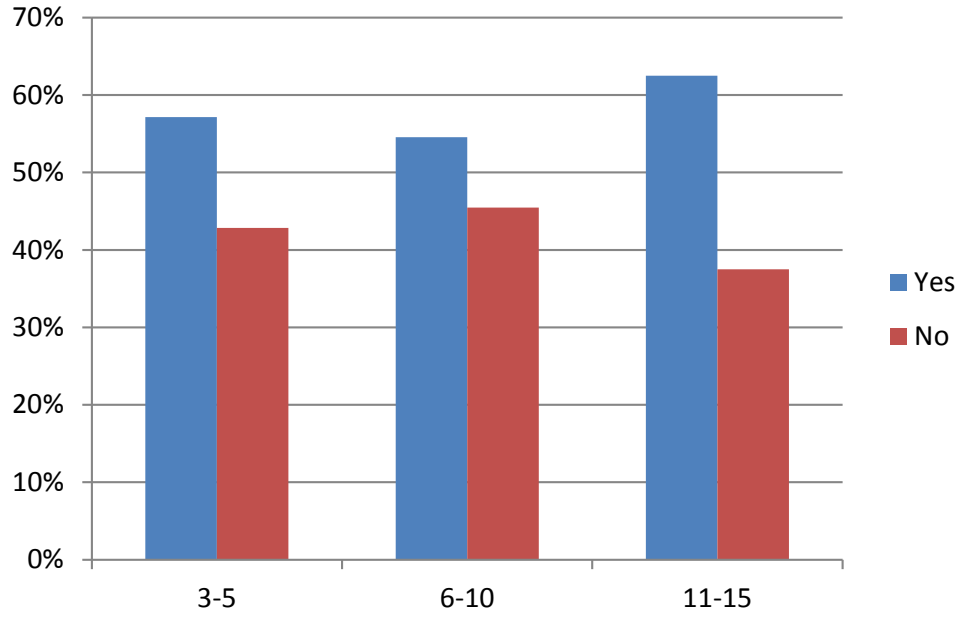

Figure 7.6: The Comparison of Pattern Writing in Merged Group

5. Overall Profile of Usefulness

All 206 respondents answered this question. But as noted above the respondents who had less than three years experience with using design patterns were not included after the comparison in the previous section. At the beginning of the survey we set the target of obviously respondents who were the experts on using design patterns at least 3 years. So after removing the data from the respondents with less than three years experience, there were 176 respondents left. Appendix I provides a summary of their ratings and their frequency in each group.

If we respectively aggregate the numbers in the two columns VU and U ('votes' representing a positive view about a pattern) and the numbers in the two columns NVU and NU ('votes' representing a negative view about a pattern) then we get the approval and disapproval tendency for each pattern. Figure 7.7 displays the votes tendency for all design patterns. The blue column represents the result of the positive aggregation. The brown column represents the result of the negative aggregation. The red horizontal line represents 50% of the respondents. From this figure we can identify a number of patterns that fail to obtain the 'approval' of more than 50% of the respondents. These are *Memento*, *Flyweight*, *Interpreter*, *Prototype*, and *Builder*. Meanwhile, the negative line displays the result for the negative 'votes'.

There are 18 patterns that obtained positive approval from more than 50% respondents. Within these patterns a small number of patterns are clearly very well known and liked, for example, Composite and Observer. While Singleton is marginally the most widely-known, the proportion of negative votes also indicates that opinions about its value are rather more mixed, particularly among the respondents with greater experience.

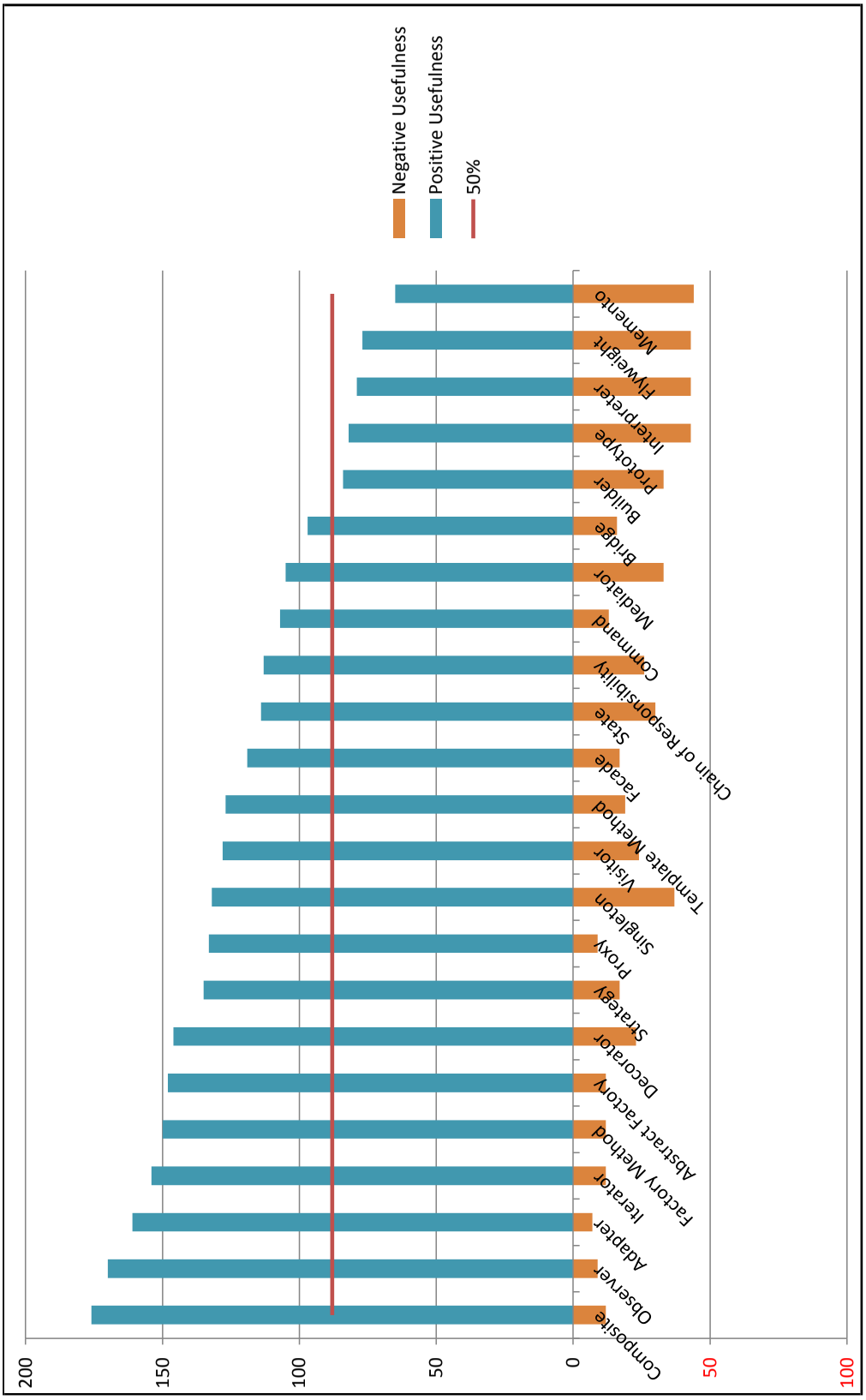

Figure 7.7: Vote Tendency in Question 7

7.4.1.2 Part II: Patterns Considered Useful

The second section (Questions 8 to 25) of this questionnaire was designed to investigate which patterns were considered useful by the respondents. It asked the respondents to select up to three patterns that they considered to be the most useful based on their experiences, and also asked them to provide the details about the type of software, system size, abstraction level, and system life-cycle involved when they applied these design patterns.

After closing the survey, 164 respondents provided some information in response to these questions. Those not responding were roughly proportional to the group sizes (25 authors, 17 from the merged group). Of those who did respond on this section, 135 provided a choice of three patterns, 11 chose two patterns, and 18 chose just one pattern. As the size of the merged group after removing those not responding was small we did not attempt to analyse the responses in two separated groups, we treated the respondents as being a single group. And also we did not consider the ordering of choice where they selected more than one pattern. Table 7.23 lists the eight most favoured patterns (to be included, a pattern had to have a total count of at least thirty across the three choices), ordered by the total number choosing that pattern as one of their three preferences.

Once again, *Observer* is very highly rated while there seems to be some hesitation about *Singleton*, as this is distinguished by mainly picking up 'third votes'. As described in the GoF, the 23 design patterns are classified into three categories, namely creational (C), structural (S) and behavioural (B), according to their purpose (Gamma *et al.*, 1995). Each pattern belongs to one of the categories. From this table there appears to be no distinction between the types of pattern preferred, all three categories are equally represented.

1. Software Types

Question 9, 15 and 21 were designed to investigate the type of software involved when the respondents made use of that pattern. Table 7.24 summarises

Pattern	Choice 1	Choice 2	Choice 3	Total
Observer (B)	24	17	13	54
Composite (S)	25	9	12	46
Abstract Factory (C)	15	9	7	31
Facade (S)	7	12	7	26
Proxy (S)	8	8	10	26
Visitor (B)	8	14	4	26
Iterator (B)	4	11	8	23
Singleton (C)	6	4	12	22

Table 7.23: Most Highly Favoured Patterns

the responses of the software type when the respondents applied the patterns. Because these questions were designed as multi-choice questions, so the total responses were significantly larger than the whole number of respondents. In this table the 23 design patterns are ranked in ascending order according to the total responses numbers. From the comparison between the different software types, there is little difference between the five types, *Productivity/Business Software*, *Graphic Design and Multimedia*, *Home/Personal/Education*, *Distributed systems*, and *System Software*. The types *Productivity/Business Software* and *Distributed systems* were the most frequently used types when the respondents employed design patterns.

	Productivity/Business Software			Graphic Design and Multimedia			Home/Personal/Education			Distributed systems			System Software			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Observer	11	6	6	9	3	5	6	7	4	9	9	7	7	3	6	98
Composite	11	4	5	7	1	5	10	3	5	6	5	5	6	1	3	77
Abstract Factory	7	3	4	2	4	0	3	3	1	6	3	2	4	4	2	48
Iterator	1	6	5	1	4	2	2	5	4	1	4	5	1	2	2	45
Façade	3	5	2	0	2	1	0	4	2	5	6	6	4	2	2	44
Proxy	0	2	3	0	0	0	0	1	2	7	5	4	5	6	1	36
Factory Method	1	5	3	1	1	1	4	0	1	3	5	3	2	1	4	35
Strategy	4	3	5	1	0	0	3	0	2	3	1	5	2	2	1	32
Singleton	5	3	4	0	1	1	1	1	4	3	1	2	2	1	2	31
Visitor	3	5	0	0	3	0	1	6	3	1	2	0	2	4	0	30
Adapter	0	2	1	1	0	3	3	0	2	2	4	2	0	3	4	27
Command	2	4	5	1	2	2	0	0	1	0	2	2	1	2	2	26
Template Method	4	2	1	0	1	1	2	1	2	4	4	1	1	1	1	26
Decorator	0	2	3	0	2	0	2	1	1	0	1	2	0	0	2	16
Bridge	1	1	0	3	0	0	1	2	0	1	0	1	1	1	0	12
Builder	1	0	0	1	0	1	0	0	0	1	1	1	0	1	2	9
State	0	2	1	1	0	1	2	2	0	0	0	0	0	0	0	9
Mediator	3	1	0	0	0	0	0	1	0	1	0	0	1	0	0	7
Chain of Responsibility	1	0	1	0	0	0	0	0	1	2	0	0	0	0	1	6
Flyweight	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	3
Interpreter	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	2
Prototype	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Memento	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Total	164			77			112			158			109			620

Table 7.24: Types of Software with Patterns Considered Useful

2. System Size

Questions 10, 16 and 22 asked the respondents to describe the size of system using the patterns selected as useful . Table 7.25 summarises the different frequencies for system size. From this table the respondents' experiences were mainly concentrated on small systems (up to 100 KLOC). There were 188 respondents who had the positive experiences of using the patterns on systems below 100 KLOC. Systems with a size of above 500 KLOC was the second most common size, and 80 respondents had experiences on tasks of this size. The size of systems between 100 to 250 KLOC is quite similar to the systems above 500 KLOC, which had 65 responses on it. Within the size range of the systems, the systems between 250 to 500 KLOC were the least common size of systems for which the respondents employed patterns.

	Up to 100			100-250			250-500			above 500			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Observer	9	10	6	8	0	2	2	2	2	4	3	3	51
Composite	14	3	8	2	1	0	2	4	0	5	1	4	44
Abstract Factory	4	3	2	3	3	2	2	2	0	6	1	3	31
Facade	4	8	3	1	2	2	0	2	0	2	0	1	25
Visitor	4	9	2	3	0	0	0	1	2	1	3	0	25
Proxy	5	2	4	2	1	1	0	1	1	1	4	2	24
Iterator	3	6	5	0	2	0	0	1	0	0	2	3	22
Singleton	1	2	9	2	0	2	0	1	1	3	0	0	21
Factory Method	2	4	3	1	2	1	0	0	1	2	2	1	19
Strategy	3	2	4	0	1	3	1	0	0	2	2	1	19
Adapter	3	2	3	0	2	0	1	0	4	0	1	1	17
Template Method	3	3	3	1	0	1	1	0	0	1	2	1	16
Command	1	1	2	1	1	2	0	0	1	0	2	1	12
Bridge	1	3	1	1	1	1	0	0	0	1	0	0	9
Decorator	2	3	1	0	1	0	0	0	0	0	0	2	9
State	2	2	0	0	1	0	0	0	0	0	1	2	8
Builder	1	0	2	0	0	0	0	1	0	1	0	0	5
Chain of Responsibility	0	0	1	1	0	1	0	0	0	1	0	0	4
Mediator	0	1	0	2	0	0	0	0	0	1	0	0	4
Flyweight	2	0	0	0	0	0	0	0	0	0	0	0	2
Interpreter	1	0	0	0	0	1	0	0	0	0	0	0	2
Prototype	0	0	0	0	0	0	1	0	0	0	0	0	1
Memento	0	0	0	0	0	0	0	0	0	0	0	0	0
Total	188			65			37			80			370

Table 7.25: System Size with Patterns Considered Useful

3. Level of abstraction

Questions 11, 17 and 23 asked the respondents to describe the level of abstraction involved in using design patterns. Table 7.26 summarises the numbers of the respondents' experiences of abstraction level in each question. From this table most of the respondents have the experiences for both of coding and design when they applied their favorite patterns. There were 238 responses that selected both coding and design to define the source of their experiences. Compared between the single choices design and code, design was the preferred choice. 87 responses described their experiences as being at design level, whereas 53 responses were at the code level when they experienced using patterns.

	Design			Code			Both			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Observer	7	9	5	1	2	0	16	5	8	53
Composite	6	1	3	2	2	2	16	6	7	45
Abstract Factory	4	1	0	3	0	1	8	8	6	31
Facade	1	5	1	1	0	1	5	7	5	26
Proxy	2	1	3	0	0	1	6	7	5	25
Visitor	0	2	1	5	3	0	3	8	3	25
Iterator	2	1	3	0	3	3	1	7	2	22
Singleton	2	0	2	0	0	3	4	4	6	21
Factory Method	0	1	0	2	1	1	3	6	6	20
Strategy	1	0	0	0	1	2	6	4	5	19
Adapter	2	1	2	1	0	0	2	4	6	18
Template Method	0	0	1	1	3	0	5	3	4	17
Command	0	2	4	0	0	0	2	2	3	13
Decorator	1	1	1	0	2	0	1	2	2	10
Bridge	1	1	1	0	1	0	1	2	1	8
State	0	2	1	0	1	0	2	1	1	8
Builder	1	0	0	1	0	1	0	1	1	5
Chain of Responsibility	0	0	0	0	0	1	2	0	1	4
Mediator	0	0	0	1	0	0	2	1	0	4
Interpreter	0	0	1	0	0	0	1	0	0	2
Flyweight	0	0	0	0	0	0	1	0	0	1
Prototype	0	0	0	0	0	0	1	0	0	1
Memento	0	0	0	0	0	0	0	0	0	0
Total	87			53			238			378

Table 7.26: Level of Abstraction with Patterns Considered Useful

4. Life Cycle of System

Questions 12, 18 and 24 asked the respondents to describe what stages in the life-cycle of the system they had applied design patterns. Table 7.27 provides the comparison between both of the stages, development and maintenance, in the life cycle of the system according to the respondents' experiences. This table shows very obviously that the development stage was the most frequent stage when the respondents applied design patterns. The total of 351 responses shows that this had been predominantly through the development of systems rather than through maintenance activities.

7.4.1.3 Part III: Patterns Considered Not Useful

In the same way as Part II, the third part (Questions 26 to 43) of this questionnaire was designed to investigate which patterns were select considered to be not useful specifically by the respondents. It also asked the respondents to select up to three such patterns based on their experiences, and also asked them to provide the details about the software type, system size, abstraction level, and system life-cycle when they applied these design patterns.

In designing the survey, finding suitable and unambiguous wording for this part required some care. The intent was to identify those patterns that respondents would actively avoid using. Far fewer of our respondents answered this part, which reflects the responses to Question 7. 29 respondents listed three patterns, 9 listed two and 20 listed only one. Of these 58 respondents, only four had less than three years experience. Based upon these responses, Table 7.28 lists the four patterns that were considered the ones to be most avoided.

When comparing this with Table 7.23 it is surprising to find *Visitor* and *Singleton* listed here, as they both feature in Table 7.23. However, this could well reflect the degree to which the successful use of some patterns is particularly dependent upon context. Indeed, when we examine the experiences from the mapping study we observe that the findings from the experimental and observation studies also provide some contradictory views about these patterns. In the

	Development			Maintenance			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Observer	23	15	13	1	1	0	53
Composite	24	7	10	1	2	2	46
Abstract Factory	15	8	7	0	1	0	31
Facade	7	11	6	0	1	1	26
Visitor	5	13	4	3	0	0	25
Proxy	8	8	7	0	0	1	24
Singleton	5	4	11	1	0	1	22
Iterator	3	9	6	0	1	2	21
Factory Method	5	7	6	0	1	1	20
Strategy	7	5	7	0	0	1	20
Adapter	5	5	6	1	0	2	19
Template Method	5	6	5	1	0	0	17
Command	2	4	7	0	0	0	13
Decorator	2	5	3	0	0	0	10
Bridge	3	3	2	0	1	0	9
State	1	4	1	0	0	1	7
Builder	2	1	2	0	0	0	5
Mediator	3	1	0	0	0	0	4
Chain of Responsibility	2	0	1	0	0	0	3
Interpreter	1	0	1	0	0	0	2
Flyweight	1	0	0	0	0	0	1
Prototype	1	0	0	0	0	0	1
Memento	0	0	0	0	0	0	0
Total	351			28			379

Table 7.27: Life-Cycle of the System with Patterns Considered Useful

Pattern	Choice 1	Choice 2	Choice 3	Total
Flyweight (B)	14	5	2	21
Singleton (C)	11	2	1	14
Visitor (B)	4	4	3	11
Memento (B)	2	6	1	9

Table 7.28: Patterns Considered Not Useful

case of Singleton we might also note that in the answers to Question 7 it was evident that the pattern was perhaps better described as being well known than valued. A discussion about individual patterns is provided in the next chapter.

1. Software Types

Questions 27, 33 and 39 were designed to investigate the software type when the respondents developed or maintained with the pattern which were considered as not useful from experience of using patterns. Table 7.29 summarises the responses for the type of software when the respondents applied the patterns. Although some respondents did not complete this part of the survey and collected data was not too much, the result of Table 7.29 can be seen as a complementary of Table 7.24. From the comparison between the different software types, there are no differences between the five types, *Productivity/Business Software*, *Graphic Design and Multimedia*, *Home/Personal/Education*, *Distributed systems*, and *System Software*. Therefore the type of systems did not cause seem to influence the negative experiences of using design patterns.

	Productivity/Business Software			Graphic Design and Multimedia			Home/Personal/Education			Distributed systems			System Software			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Singleton	6	1	1	2	1	0	2	1	0	4	1	1	2	1	1	24
Flyweight	0	1	1	2	0	2	3	0	1	2	2	1	3	0	1	19
Visitor	1	0	1	0	1	2	0	1	1	1	1	2	1	1	1	14
Facade	0	1	0	1	1	1	0	1	1	0	2	0	0	2	0	10
Chain of Responsibility	1	1	1	0	0	1	0	1	1	0	0	1	0	0	1	8
Template Method	1	0	0	1	0	0	2	0	0	1	0	0	2	0	0	7
Interpreter	1	0	0	0	0	1	0	0	1	1	0	0	0	0	1	5
Bridge	2	0	0	0	0	0	0	0	0	1	0	0	1	0	0	4
Iterator	0	1	0	0	1	0	0	1	0	0	0	0	0	1	0	4
Memento	0	0	0	0	1	0	1	1	1	0	0	0	0	0	0	4
Prototype	0	1	0	0	0	0	1	0	0	0	1	0	0	1	0	4
Strategy	1	0	0	1	0	0	0	0	0	0	0	1	0	0	1	4
Composite	0	0	0	0	1	0	0	1	0	0	0	0	0	1	0	3
Proxy	0	1	0	1	0	0	1	0	0	0	0	0	0	0	0	3
Builder	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	2
Command	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	2
Mediator	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	2
Observer	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	2
State	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	2
Abstract Factory	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1
Decorator	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Adapter	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Factory Method	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Total	28			24			25			25			23			125

Table 7.29: Types of Software with Patterns Considered Not Useful

2. System Size

Questions 28, 34 and 40 asked the respondents to describe the size of system for which they had the worst experiences of using patterns. Table 7.30 summarises the different frequencies of the system size. Although some respondents did not complete this part of the survey and collected data was not too much, the result of Table 7.30 can be seen as a complementary of Table 7.25. From this table the respondents' experiences were mainly concentrated on small systems (up to 100 KLOC). There were 24 respondents who had negative experiences of using patterns on system below 100 KLOC in size. Systems of 100-250 KLOC were the second most common size, and 21 respondents had experiences with this size. The number of systems above 500 KLOC is quite similar to the systems of size 100-250 KLOC, which had 18 responses. Within the size range for systems, those between 250 to 500 KLOC were the least common size for systems the respondents worked on when they applied these patterns.

	Up to 100			100-250			250-500			above 500			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Mediator	4	1	0	2	0	1	0	0	0	3	0	0	11
Adapter	2	1	0	2	1	1	0	0	0	1	0	1	9
Memento	1	1	1	1	1	0	0	0	0	1	1	0	7
Visitor	0	1	1	0	0	1	1	1	0	0	0	0	5
Factory Method	0	1	0	0	1	1	0	0	0	1	0	0	4
Proxy	1	0	0	0	0	0	0	0	1	1	0	0	3
Template Method	0	0	1	1	0	0	0	0	0	1	0	0	3
Decorator	1	0	1	0	1	0	0	0	0	0	0	0	3
State	0	0	0	0	0	1	0	0	0	1	1	0	3
Chain of Responsibility	1	0	0	1	0	0	0	1	0	0	0	0	3
Abstract Factory	0	0	0	1	0	0	1	0	0	0	0	0	2
Command	0	1	0	0	0	0	0	0	0	0	1	0	2
Bridge	0	0	1	0	0	0	0	0	0	0	1	0	2
Builder	0	0	0	0	1	0	0	0	0	1	0	0	2
Flyweight	0	0	0	0	1	0	0	1	0	0	0	0	2
Interpreter	0	0	0	0	0	1	0	0	0	1	0	0	2
Prototype	2	0	0	0	0	0	0	0	0	0	0	0	2
Observer	0	0	0	0	0	0	0	0	0	0	0	1	1
Facade	0	0	0	0	0	0	0	0	0	0	0	1	1
Iterator	0	1	0	0	0	0	0	0	0	0	0	0	1
Singleton	0	0	0	0	1	0	0	0	0	0	0	0	1
Composite	0	0	0	0	0	0	0	0	0	0	0	0	0
Strategy	0	0	0	0	0	0	0	0	0	0	0	0	0
Total	24			21			6			18			69

Table 7.30: System Size with Patterns Considered Not Useful

3. Level of abstraction

Questions 29, 35 and 41 asked the respondents to describe the level of abstraction involved in using these design patterns they considered not useful. Table 7.31 summarises the numbers of the respondents' experiences of the level of abstraction in each question. Although some respondents did not complete this part of the survey and collected data was not too much, the result of Table 7.31 can be seen as a complementary of Table 7.26. From this table most of the respondents have experiences from both of coding and design when they employed the least favorite patterns. There were 38 responses that selected both coding and design as defining their experiences. Compared between the single choices of design and code, the respondents preferred the design choice. 24 responses described their experiences as design level, 9 responses were at the code level.

	Design			Code			Both			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Mediator	2	0	0	2	0	0	5	1	1	11
Adapter	1	1	1	1	1	0	3	1	1	10
Memento	1	2	1	1	0	0	1	1	1	8
Proxy	0	1	1	0	0	0	1	1	1	5
Visitor	1	0	0	1	0	0	1	0	1	4
Factory Method	0	1	0	1	0	0	0	1	1	4
Template Method	2	0	0	0	0	0	0	0	1	3
Bridge	1	1	1	0	0	0	0	0	0	3
State	1	0	0	0	0	0	0	1	1	3
Chain of Responsibility	1	0	0	0	0	0	1	1	0	3
Abstract Factory	1	0	0	0	0	0	1	0	0	2
Command	0	0	0	0	1	0	0	1	0	2
Decorator	0	0	1	0	0	0	0	1	0	2
Builder	0	1	0	0	0	0	1	0	0	2
Interpreter	0	0	0	0	0	0	0	2	0	2
Flyweight	0	0	0	1	0	0	0	0	1	2
Observer	0	0	0	0	0	0	0	0	1	1
Facade	0	0	0	0	0	0	0	0	1	1
Iterator	0	0	0	0	0	0	0	1	0	1
Singleton	0	1	0	0	0	0	0	0	0	1
Prototype	0	0	0	0	0	0	1	0	0	1
Composite	0	0	0	0	0	0	0	0	0	0
Strategy	0	0	0	0	0	0	0	0	0	0
Total	24			9			38			71

Table 7.31: Level of Abstraction with Patterns Considered Not Useful

4. Life Cycle of System

Questions 30, 36 and 42 asked the respondents about the stage in the life-cycle of the system when they used the design patterns which they considered not useful. Table 7.32 provides a comparison between both of the stages, development and maintenance. Although some respondents did not complete this part of the survey and collected data was not too much, the result of Table 7.32 can be seen as a complementary of Table 7.27. Similarly to the positive views, this table shows very obviously that the development stage was the most frequent stage when the respondents applied design pattern. The number of 64 responses shows that this was predominantly through the development of systems rather than through maintenance.

7.4.1.4 The Combined Responses

Figure 7.8 shows the counts for each pattern between Question 8 to Question 42 which are designed for investigating the details of the single patterns. Those columns above the line are the aggregated votes for the pattern as being useful in the three questions 8, 14, 20. Those below are votes for the same pattern as not being useful in the questions 26, 32 and 38. Figure 7.9 adopts the percentage to assess the usefulness of design patterns. It shows the percentage of responses which considered useful and not useful for each pattern in the whole responses. Compared with Figure 7.8 both of them shows the same distributions. And Figure 7.9 displays the conflict responses for each pattern very clearly.

Combined this figure with Figure 7.7, the order of the patterns considered to be most useful and least useful patterns are quite similar. But still there are some interesting patterns that stand out. This visualisation does particularly highlight the ambivalence about Singleton and Visitor patterns. After comparing with Figure 7.7 we find that the Singleton pattern appears in both highly useful and highly not useful patterns, and that Visitor also appears in both of them. In the next section we therefore concentrate upon categorising the comments that were made about these two patterns.

	Development			Maintenance			Total
	Choice 1	Choice 2	Choice 3	Choice 1	Choice 2	Choice 3	
Chain of Responsibility	5	1	1	5	0	0	12
Adapter	5	2	2	0	1	0	10
Memento	3	2	2	0	1	0	8
Visitor	1	2	2	0	0	0	5
Proxy	3	0	1	0	0	0	4
Factory Method	0	2	1	1	0	0	4
Template Method	2	0	1	0	0	0	3
Bridge	1	1	1	0	0	0	3
State	1	1	1	0	0	0	3
Mediator	2	1	0	0	0	0	3
Abstract Factory	2	0	0	0	0	0	2
Command	0	2	0	0	0	0	2
Decorator	0	1	1	0	0	0	2
Builder	1	1	0	0	0	0	2
Interpreter	0	2	0	0	0	0	2
Flyweight	1	0	1	0	0	0	2
Observer	0	0	1	0	0	0	1
Facade	0	0	1	0	0	0	1
Singleton	0	1	0	0	0	0	1
Iterator	0	1	0	0	0	0	1
Prototype	1	0	0	0	0	0	1
Composite	0	0	0	0	0	0	0
Strategy	0	0	0	0	0	0	0
Total	64			8			72

Table 7.32: Life-Cycle of the System with Patterns Considered Not Useful

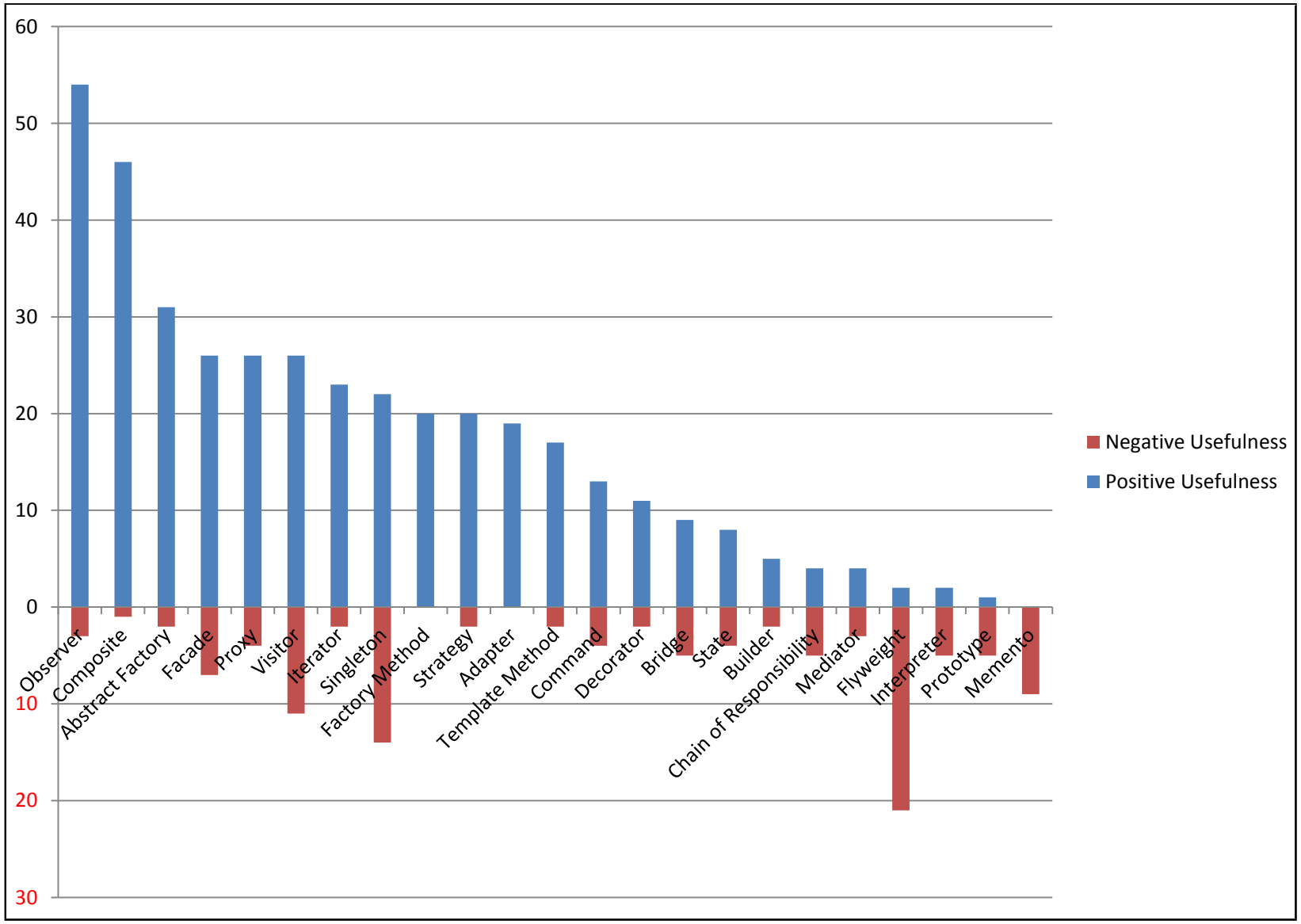

Figure 7.8: Combined Responses Comparison from Question 8 to Question 42

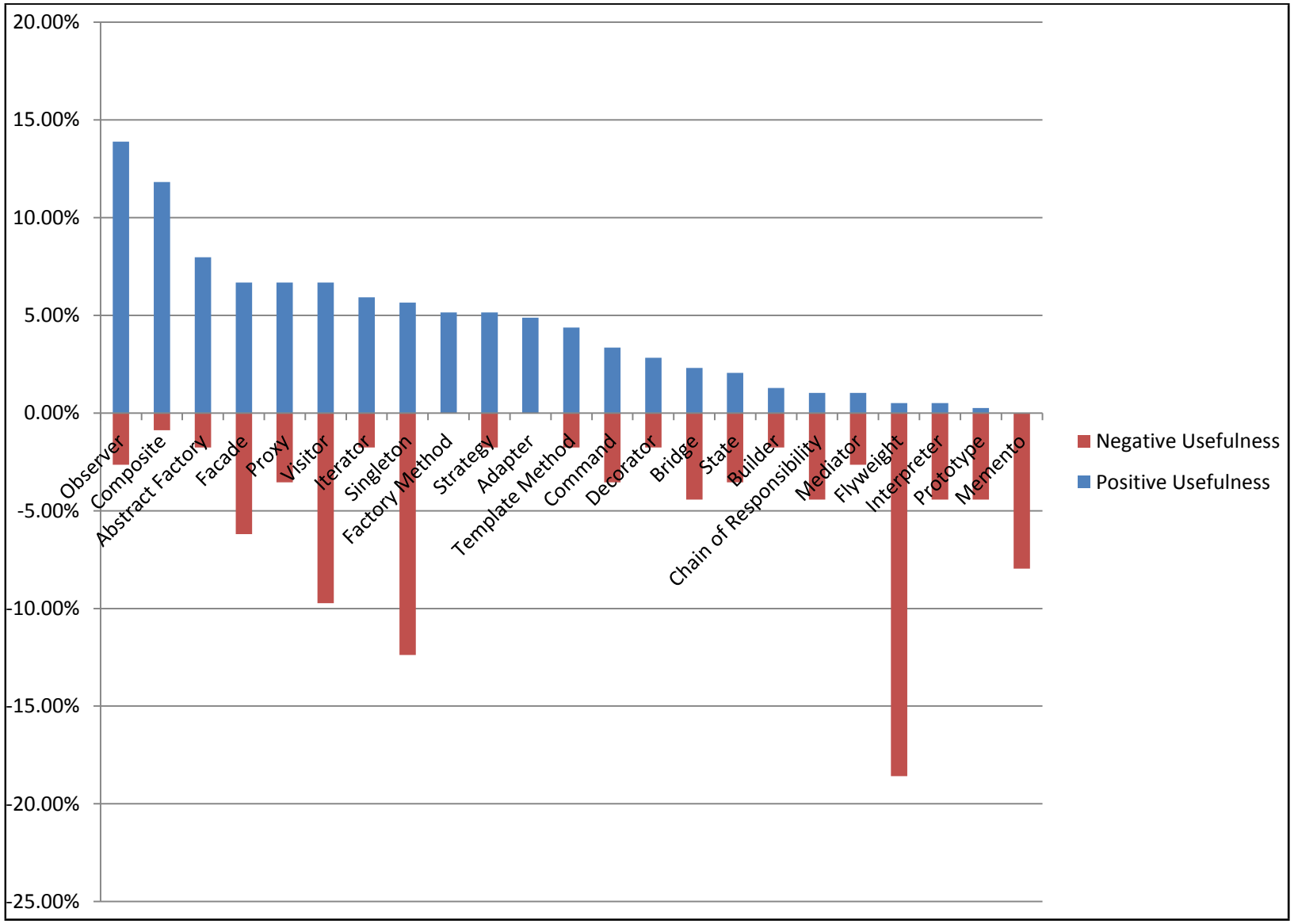

Figure 7.9: Combined Responses Percentage Comparison from Question 8 to Question 42

7.4.2 Qualitative Data

Qualitative data is not numerical data (Seaman, 1999). Questions 13, 19, 25, 31, 37, 43 in the survey were designed as open-ended questions. The participants were expected to provide comments to share their experiences about using those design patterns they considered useful and not useful. In this section we provide a qualitative analysis of the data from these questions. We focus especially on the patterns *Visitor* and *Singleton*, since these two patterns appear in both Table 7.23 and Table 7.28, which display the patterns that are considered useful and not useful respectively.

In the survey participants were asked to describe their experiences and the characteristics of the design patterns in simple words and brief phrases. A qualitative analysis sorts and sifts the responses, and reconstructs and finds the data in a meaningful way (Jorgensen, 1989). The answers from these open-ended questions are all brief text. The analysis is based on the qualitative analysing process described in (Seaman, 1999; Taylor-Powell & Renner, 2003).

7.4.2.1 Categorisation and Coding

There were 20 participants who commented on the *Visitor* pattern when reporting positive experiences in Questions 13, 19, and 25. And also 8 participants who commented on this pattern for negative experiences in Questions 31, 37, and 43. For the *Singleton* pattern, 19 participants described it for positive experiences, and 8 participants provided negative experiences.

These comments were categorised according to three criteria: participants' experiences, software quality, and Object-Oriented issues. Because these comments are from the participants' experiences, the categories should be based on the experiences. Table 7.33 presents the examples of the categories. We coded each response using a limited set of words for these categories. These words were generated from Questions 11 and 12 of the survey, which were used to investigate the level of abstraction and life-cycle of the system in their experiences.

After categorising the qualitative data, each participant's comment was coded.

Questions	Categories
What was the level of abstraction involved in using this design pattern?	Code / Design/ Both
In what stage(s) in the lifecycle of the system(s) did you work with this design pattern?	Development / Maintenance

Table 7.33: Qualitative Data Categories

In the coding process we sought the special perspectives and meanings from each participant's experience, rather than generalising the comments simply (Taylor-Powell & Renner, 2003). These coding key words can also be used for the interpretation.

7.4.2.2 Interpretation

1. The Visitor Pattern

> "*Represents an operation to be performed on the elements of an object structure. Visitor lets you define a new operation without changing the classes of the elements on which it operates.*" (Gamma *et al.*, 1995)

20 participants who rated this among their 'top three' provided comments, and 8 who rated it under their 'not desirable' choices provided comments. Most people considered it as a useful pattern in at both code and design level. During both the development and maintenance process it should aid with making a system extensible and maintainable. It is a great tool for decomposition and decoupling. In the maintenance process the most useful characteristic of the pattern is encapsulation, and it is good for applying in web development. Besides that it is also useful in the context of language processors. Few responses really gave details

about experiences in support of their views, but some positive views that did so include:

- "We conduct different analyses on Abstract Syntax Trees. Visitor can provide a unique way to do them and operations are completely independent."
- "The ability to create many visitors for the same data model. This is very useful for web development. This combined with the MVC pattern means that we can create a lot of views with ease. For example, we have an html table printer, csv table printer etc. for the same table of information."
- "The visitor is very useful in the context of language processors. I have used it primarily to support AST/ASG traversals. The time and effort involved in modifying a visitor hierarchy can be prohibitive, but these factors are balanced by its natural suitability for tree/graph traversals."

But due to its properties of abstract and complication, it is a not a well-understand pattern. Its use requires extensive experience of software development. Otherwise it is hard to manage and maintain, especially for novices. And also it is easy to lead to a poor structure. Sometimes it costs very prohibitive time and effort in system design.

- "I prefer to use multiple dispatch, however in systems without multiple dispatch you have to emulate it with a visitor. The resulting code using visitor is easy to get wrong, hard to maintain, and difficult to understand."
- "Only useful to ship data structures and algorithm separately. But then, you have to fix either a set of data structures or a set of algorithms (depending on who visits who). So it can be a pain to manage."
- "To avoid procedural dependencies of conditionals, I prefer to use idioms that utilize polymorphism to resolve the state of conditionals and the message to send as a consequence of that state. The visitor pattern requires too much awareness of handshaking to be practical. Supporting implementation details needs to be invisible so they don't distract from the focus of the role being designed, and provide less opportunity for defects to be injected into the system."

2. The Singleton Pattern

> "*Ensure a class only has one instance, and provide a global point of access to it.*" (Gamma *et al.*, 1995)

For this pattern we had 19 comments from those who rated it highly and 8 who recorded having negative experiences with its use. Most of the respondents applied it in both of code and design level. In the design process people think applying Singleton pattern can make design easily. The most common characteristic of the Singleton pattern is it prevents a client creating more than one instance of a specific class, it provides for good control of object creation and avoids duplication of similar objects. And for maintenance the Singleton pattern makes a system easier for learning and using, its use should improve system performance.

The Singleton pattern is a very controversial pattern. Even Eric Gamma (Gamma *et al.*, 1995), one author of GoF, said in an interview that he was in favour of dropping Singleton and the use of it was always a "design smell" [1]. From the comments of the survey participants, the Singleton pattern is considered as massively misused and overused to provide global variables. Some even considered it as an anti-pattern. Some people think Singleton also produces bad coupling.

Two positive views of Singleton were:

- "Singleton is natural for use in logging libraries and to maintain user configurations. It provides the convenience of a global variable (without the dirty feeling)."
- "Singleton is a basic pattern, but still it needs some discipline. In code analysis, global entities (e.g. symbol table) may need to have a single instance."

While the opposing views included:

- "I have rarely the need of a singleton as most domain objects are not unique. In maintenance, this is often used to hold global variables."

[1] http://www.informit.com/articles/article.aspx?p=1404056

- "Singleton is more an anti-pattern and introduces global state."
- "This pattern introduces 'temporal coupling' (the worst kind). I NEVER use this pattern. The last time I saw it and had to deal with it was 5 years ago and it was extremely painful to retrofit unit tests in that project, because of the Singletons."

7.5 Summary

In this chapter we analysed the response data from the online survey. These data were classified based the types in the survey. Then they were analysed by the quantitative method and the qualitative method. In the quantitative analysis we employ the descriptive statistics and the inferential statistics to analyse the data. For the inferential statistics we applied the hypothesis testing to analyse the relations between the respondents' profiles and the pattern application. To analyse the specific patterns which were considered useful or not useful, we mainly employed the descriptive statistics with supporting of SPSS. At the end of the analysis, we also analysed the answers of the open questions by using the qualitative method, though we received relatively little in the way of comments.

The analysing results indicate that use of patterns is related to users' experiences. Of the specific patterns the Composite, Observer and Abstract Factory patterns are considered the most useful patterns. Conversely, the Memento, Flyweight patterns are regarded as the least useful patterns. We analysed the specific patterns from the aspects of software type, system size, level of abstraction, and life-cycle stage. Within the patterns being considered most useful and least useful, the Visitor and Singleton patterns are found to be the most controversial patterns. They appear in the both of the most useful patterns and least useful patterns lists. So we analysed these two patterns by analysing the qualitative data.

Chapter 8

DISCUSSION

8.1 Introduction

We have performed a document survey to identify claims for patterns (Chapter 2), a mapping study (Chapters 4 and 5), and the online survey (Chapters 6 and 7) as described in the previous chapters. In the claims survey we sought claims from sources such as textbooks, and those papers that investigated the use of design patterns, and used the results to check how extensively the later studies such as the mapping study and the online survey met the claims for design patterns. In the mapping study we investigated how extensively the use of software design patterns had been subjected to empirical evaluation, what evidence could be found to indicate the extent to which their use can provide an effective mechanism for knowledge transfer, and under what conditions. The online survey was then conducted to investigate the relations between the profiles of users and the design patterns, and also identify which patterns from the set catalogued by the 'Gang of Four' are considered to be useful by researchers and developers, which ones are considered as not useful, and why this is so. All of the results of the preliminary synthesis and analysis for those three studies have been presented.

This chapter aims to organise, compare and synthesise the results of the three studies by using a narrative synthesis. Narrative synthesis is a descriptive synthesis method which is tabulated in a manner consistent with the research questions

to find the similarities and differences between the outcomes of studies (Kitchenham & Charters, 2007). Here we tabulated the results of the three studies from the general aspects and the specific patterns aspects. First of all from the general aspects we compared and synthesised the results of the studies relating to some claims which were extracted from the claims survey, and some factors which were extracted from the mapping study and the online survey. These general aspects were considered as having impact on the application of all the design patterns. We compared and tried to summarise that whether there was any effect on these claims and factors arising from employing design patterns. For those claims which had obvious positive effects we considered them as the certified claims and factors. Conversely, those which had negative effects or had no effects were considered as the uncertified claims and factors. Then from the aspects measured for the specific patterns, we compared and synthesised the studies in terms of some claims and also some factors which were extracted from the mapping study and the online survey. After tabulating for synthesis, we analysed the results by using a textual description.

In this chapter we synthesised all of the evidence from the three studies and explored the relations between them to seek answers to these questions:

- How well does the available empirical data support the claims that are made for design patterns?
- Which patterns have been studied most widely?
- How does the use of design patterns influences the process of software development?

After tabulating and analysing the evidence synthesis from the general aspects and the aspects of the specific patterns, we also assessed the robustness of the synthesis by using a validity assessment.

8.2 Evidence Synthesis

This section describes the evidence synthesis process by using narrative synthesis. The forms of evidence that were collected from the mapping study, the online survey, and the claims survey were synthesised from the general aspects and the aspects of the specific patterns. Table 8.1 shows how these claims and factors are derived as the properties of patterns when viewed from different aspects such as software design, component, evaluation, developers and users. Among these the label "system size" is extracted as a factor from both the mapping study and the online survey. As described in the chapter about the online survey, the scale of the system size is based on the numbers of lines of code. A system which has less than 100 KLOC is regarded as a small system. Systems between 100 to 250 KLOC are regarded as medium systems. Systems between 250 to 500 are regarded as big system. Any systems which has more 500 KLOC is regarded as a large system. This factor was used to evaluate whether system size is relevant with the application of design patterns. So we just applied the scale "Small / Medium / Big / Large" to synthesise the studies on the system size factor.

8.2.1 General Aspects

Table 8.2 summarises the extent to which the claims and the factors are supported or otherwise by the mapping study and the online survey, and also the overall effects of the application design patterns. In this table we have used some symbols to represent the different effects. The symbol "+" represents the study provides a positive support for the claim. The symbol "-" represents the study does not support for the claim. The symbol "±" represents where study provides the conflicting values for the claim.

8.2.1.1 Certified Claims & Factors

1. Encourage Best Practices

This claim indicates that design patterns encourage best practices for designers. In other words, design patterns can help designers to choose alternative design

No.	Labels	Claims & Factors
1	Encourage best practices	"Patterns encourage best practices, even for experienced designers." (Prechelt *et al.*, 2002) "Design patterns help you choose design alternatives that make a system reusable and avoid alternatives that compromise reusability." (Gamma *et al.*, 1995)
2	Vocabulary	"Design patterns improve communication, both among developers and from developers to maintainers."(Prechelt *et al.*, 2002) "Patterns provide a common vocabulary and understanding for design principles."(Buschmann *et al.*, 1996)
3	Level of abstraction	"The benefits of design patterns include the reuse of design instead of program" (Dong *et al.*, 2007) "Design patterns don't go directly into your code, they first go into your brain." (Freeman *et al.*, 2004)
4	Life cycle	"Using these patterns early in the life of a design prevents later refactoring." (Gamma *et al.*, 1995)
5	Reuse design & architecture	"Design patterns make it easier to reuse successful designs and architectures." (Gamma *et al.*, 1995)
6	Improve productivity & quality	"Using patterns improves programmer productivity and program quality." (Prechelt *et al.*, 2002)
7	Increase skills of novices	"Novices can increase their design skills significantly by studying and applying patterns." (Prechelt *et al.*, 2002)
8	Properties	"Patterns support the construction of software with defined properties." (van Vliet, 2000)
9	Software type	"If a pattern can be found in some - say two or three - commercially or otherwise widely deployed systems, it is considered to be proven." (Kerth & Cunningham, 1997)
10	System size	Small / Medium / Big / Large
11	Abstraction	"Design patterns help you identify less-obvious abstracts and the objects that can capture them." (Gamma *et al.*, 1995)
12	Interface	"Design patterns help you define interfaces by identifying their key elements and the kinds of data that get sent across an interface." (Gamma *et al.*, 1995)
13	Documentation	"Patterns are a means of documentation." (van Vliet, 2000)

Table 8.1: Claims and Factors for Narrative Synthesis

No.	Labels	Mapping Study	Online Survey	Effect
1	Encourage best practices	+		+
2	Vocabulary	+		+
3	Level of abstraction	+	+	+
4	Life Cycle	+	+	+
5	Reuse design & architecture	±	±	±
6	Improve productivity & quality	±	±	±
7	Increase skills of novices	-	-	-
8	Properties	±	±	±
9	Software type	-	-	-
10	System size	Small	Small	Small
11	Abstraction			
12	Interface			
13	Documentation			

Table 8.2: The Result of Narrative Synthesis between Three Studies from the General Aspects

to achieve the best practices and avoid problems. This claim is supported by the overall results of the mapping study. The online survey has no input about this claim. The experiment papers FS2, FS7 from the mapping study provided some positive observations upon this claim:

> "since many of the benefits of factories can be achieved by alternative solutions that do not incur the same usability penalty, the results . . . suggest that such alternatives are often preferable to factories" (Ellis *et al.*, 2007)

> "unless there is a clear reason to prefer the simpler solution, it is probably wise to choose the flexibility provided by the design pattern solution because unexpected new requirements often occur" (Prechelt *et al.*, 2001)

As a consequence, it can be argued that design patterns can offer alternative solutions to reduce the design risk.

2. Vocabulary

This claim is derived from (Prechelt *et al.*, 2002) and (Buschmann *et al.*, 1996), and it indicates that design patterns can improve the communication between designers and designers, or designers and maintainers by providing a common vocabulary. From our mapping study there is reasonably good support for the claim that using patterns improves communication, at least when these are appropriately documented. The online survey again has no input regarding this claim. The experiment studies FS8, FS9, FS10 support this claim and provided the following positive observations:

> "depending on the particular program, pattern comment lines in a program may considerably reduce the time required for a program change or may help improve the quality of the change" (Prechelt *et al.*, 2002)

> "since our subjects were less expert and trained than (Prechelt *et al.*, 2002), our support is stronger" (Torchiano, 2002)

> "team members can communicate more effectively with design pattern knowledge" (Unger & Tichy, 2000)

Therefore, synthesising the evidence we obtained from the studies indicates that design patterns can improve the communication for designers and maintainers.

3. Level of Abstraction

This claim is derived from (Dong *et al.*, 2007; Freeman *et al.*, 2004). They indicate that design patterns should be employed in the design process rather than the programming process. This is supported by the results of the mapping study and the online survey.

In the mapping study we investigated the level of abstraction at which they used design patterns in the experience papers. Table 5.14 shows that 5 out of 7 experience papers applied patterns in design process. And in the online survey the results from Tables 7.26 and 7.31 show that, comparing with programming, the design process is the respondents' preferred choice for when they applied patterns. Thus there is support for the view that design patterns mainly provide benefit for the design process.

4. Life Cycle

This claim indicates that it is better to apply design patterns in the earlier stages of the system life-cycle. The results of the mapping study and the online survey support this claim. In the mapping study, Table 5.14 shows that 6 out of 7 experience papers employed design patterns in the development stage. And also in the online survey Tables 7.27 and 7.32 show that around 90% respondents used patterns in the development stage. Although our survey mainly sampled developers and it could be biased, the respondents of this survey were the paper authors in the mapping study, the papers of the mapping study mainly focused on the development stage. Although the respondent used design patterns in the development stage, it does not mean design patterns are not important in the maintenance stage.

5. System Size

This factor is extracted from the mapping study and the online survey. When we performed the mapping study and the online survey we tried to investigate the relation between the system size and the application of design patterns. Through observing the evidences from the mapping study and the online survey, small size seems to be preferred when design patterns are employed. In the mapping study Tables 5.10 and 5.14 show that most users applied design patterns in systems that were smaller than 100 KLOC. In the online survey Tables 7.25 and 7.30 show that 188 out of 370 respondents who had positive experiences applied patterns in small systems (up to 100 KLOC), and 24 out of 69 respondents who had negative experiences applied patterns in small systems.

As a consequence, the users' experiences of applying design patterns were mainly concentrated on the small size systems. So any conclusions only really apply to smaller systems.

8.2.1.2 Uncertified Claims & Factors

1. Reuse design & architecture

This claim is derived from the GoF textbook. It indicates that design patterns can make it easier to reuse successful design and architecture. Actually this book assumes that the readers are proficient in at least one OO programming language (Gamma *et al.*, 1995). But in our mapping study and the online survey support for this claim is less consistent.

The mapping study provided some conflicting outcomes from replicated studies, for example, FS7 and FS11 observed opposite results for applying some patterns. And also in the online survey some patterns such as the Singleton pattern and the Visitor pattern were listed in both of Table 7.23 for the most favoured patterns and Table 7.28 for the least favoured patterns. Perhaps the different conditions of using patterns influenced the results of the successful reuse.

Therefore, reusing the successful design patterns in design and architecture without considering other conditions can not determine whether we can perform a project easily or not. It can even make for a more complex design. So the conditions that affect reuse of patterns successfully are the claims and factors we investigate in the thesis.

2. Improve Productivity & Quality

This claim indicates that using patterns improves programmer productivity and program quality. But the conclusions from the mapping study and the online survey related to productivity and quality are ambivalent - arguably these issues are strongly affected by the nature of individual patterns.

With regard to this claim, for studies FS2, FS3, FS4, and FS12, the authors of these studies observed that:

> "factories are demonstrably more difficult than constructors for programmers to use, regardless of context." (Ellis *et al.*, 2007)
>
> "The experiment suggested that the satisfaction of the pattern theme generally led to the conformance to the open-closed principle. However, three exception cases were found." (Ng *et al.*, 2006)
>
> "utilizing deployed design patterns . . . is found to be statistically significantly associated with the delivery of less faulty codes" (Ng *et al.*, 2007)
>
> "(Visitor) does not reduce the subjects' efforts for comprehension tasks . . . its canonical representation reduces the developers' efforts for modification tasks" (Jeanmart *et al.*, 2009)

The online survey also reflected this. Figure 7.8 shows that no respondents considered the Factory Method pattern as a 'not useful' pattern, but conversely no respondents considered the Memento pattern as a 'useful' pattern. So to sum up the results from the mapping study and the online survey, any conclusion about the claim for productivity and quality improvement is necessarily pattern-specific.

3. Increase Skills of Novices

This claim indicates that learning and applying design patterns can promote the skills of novices. But neither our mapping study nor the online survey provided any support for this claim.

In the mapping study the experiment paper FS13 and the experience paper O7 specifically comment upon in this claim. From the negative observations made, their authors clearly did not support this claim.

> "the main difficulty faced by novices was the incorporation of patterns into the initial class diagram" (Abdul Jalil & Noah, 2007)

> "students found it difficult to relate the applied design patterns to specific design problems" (Chatzigeorgiou *et al.*, 2008)

The result of the online survey also leads to the same conclusions. Through the analysis of the survey, the application of design patterns is relevant to users' experience. Figures 7.1 and 7.2 show that those participants who have little experience of using patterns may consider patterns are useless. Even the GoF textbook also points out that book should be useful to those readers who are already proficient in OO language and have some experience in OO design (Gamma *et al.*, 1995). Therefore, to synthesise evidence about this claim, we found that design patterns cannot increase the skills of novices, and novices can face some difficulties with software design when using design patterns. This claim should be under the condition that the users should have some basic level of experience with OO design and design patterns.

4. Properties

This is a very controversial claim for the application of design patterns. This claim indicates that design pattern can help to design for defined properties. The defined properties are classified as the functional properties and the non-functional properties (Buschmann *et al.*, 1996). The functional properties deal

with the specific functions according to a system's requirements. The non-functional properties are related such aspects as reliability, cost, maintenance or development. Because the functional properties are related to the specific system's requirements, we have focused on the non-functional properties in this thesis.

In both the mapping study and the online survey, we found some controversial results about the properties of systems created by using design patterns. For example, for the Observer pattern some papers ($FS8_1$, FS11 and O5) agreed that it made design more understandable. But the authors of papers FS7 and O3 argued that it increased the time needed and also had a risk of creating an over-complicated solution. The same effects were also found from the responses of the open question in the online survey. Respondents provided quite different views about the properties arising from using the Singleton pattern and the Visitor pattern. One respondent expressed a positive view for the maintainability of systems created by using the Singleton pattern:

> "Singleton is natural for use in logging libraries and to maintain user configurations."

While another provided a quite negative view about the functionalities created by using the Visitor pattern:

> "The time and effort involved in modifying a visitor hierarchy can be prohibitive..."

Therefore, it is very difficult to determine whether design patterns can consistently provide defined properties for software design. A single pattern cannot ensure the presence of all the desired properties for a system. The results will depend on the experience of pattern users and the other development conditions.

5. Software Type

This claim indicates that design patterns can improve reuse in certain types of system. However this claim is not supported by the results of our studies. The

results do not have any effect on this claim.

The evidence from both of the mapping study and the online survey show that there is no relation between software type and the patterns application. Table 5.14 shows that the experience papers applied design patterns in a variety of systems, including business system and education system. But the effectiveness of design patterns did not change with the system type. The observation from the online survey supported this observation. Tables 7.24 and 7.29 show that there was no difference between the five types of system, *Productivity/Business Software*, *Graphic Design and Multimedia*, *Home/Personal/Education*, *Distributed systems*, and *System Software*, when the respondents applied design patterns. Therefore, the successful reuse of design patterns is not dependent upon software type.

6. Abstraction / Interface / Documentation

The results of the mapping study and the online survey do not have any effect on these three claims. These claims for design patterns are also entwined with a definition of what a pattern is. For example, "patterns are a means of documentation" is taken from (van Vliet, 2000) and is really a definition of the role of a pattern. And also the claims "design patterns help you identify less-obvious abstracts and the objects that can capture them " and "design patterns help you define interfaces by identifying their key elements and the kinds of data that get sent across an interface " are taken from the GoF and they mainly focus on the programming issues. These issues are determined by the different conditions of system design. So their validity are quite difficult to assess from the mapping study and the online survey.

8.2.2 Aspects of Specific Patterns

We have synthesised the general aspects of using design patterns. But the GoF textbook contains 23 design patterns. Each pattern will have its own characteristics related to the claims and factors such as *Software type*, *System size*, *Level of*

abstraction, and *Life cycle* which have been described in Table 8.1. We counted the claims frequencies from Tables 7.24, 7.25, 7.26, and 7.27 for each pattern considered useful in the online survey. Table 8.3 shows the most preferred values for these items for each pattern. From this table, although the claim '*Software type*' has no relation with using design patterns as described previously, each pattern still has its own most appropriate software type. And also these patterns were mainly applied in small systems development. For the claim '*Level of abstraction*', all patterns were concentrated on both of the design and coding processes. This option in the online survey provided the respondents with a flexible choice, so that the respondents can describe their experiences by choosing this option when they can not remember clearly. So we also had to compare between the design process and the coding process as described in the previous section. The respondents' experiences about using patterns were mainly concentrated on the design process. From Table 8.3 all of the design patterns were used in the development stage. In addition, we should also note that the Memento pattern has no preferred items in the claims and factors, since no respondents considered it to be a useful pattern.

Patterns	Software Type	System Size	Level of Abstraction	Life Cycle
Abstract Factory	Productivity/Business Software	above 500	Both	Development
Adapter	Distributed system	Up to 100	Both	Development
Bridge	Graphic Design and Multimedia & Home/Personal/Education	Up to 100	Both	Development
Builder	Distributed systems & System Software	Up to 100	Both	Development
Chain of Responsibility	Productivity/Business Software & Distributed system	100-250	Both	Development
Command	Productivity/Business Software	Up to 100 & 100-250	Both	Development
Composite	Productivity/Business Software	Up to 100	Both	Development
Decorator	Productivity/Business Software	Up to 100	Both	Development
Facade	Distributed system	Up to 100	Both	Development
Factory Method	Distributed system	Up to 100	Both	Development
Flyweight	Graphic Design and Multimedia	Up to 100	Both	Development
Interpreter	Distributed system & System Software	Up to 100 & 100-250	Both	Development
Iterator	Productivity/Business Software	Up to 100	Both	Development
Mediator	Productivity/Business Software	100-250	Both	Development
Memento				
Observer	Distributed system	Up to 100	Both	Development
Prototype	Productivity/Business Software	250-500	Both	Development
Proxy	Distributed system	Up to 100	Both	Development
Singleton	Productivity/Business Software	Up to 100	Both	Development
State	Home/Personal/Education	Up to 100	Both	Development
Strategy	Productivity/Business Software	Up to 100	Both	Development
Template Method	Distributed system	Up to 100	Both	Development
Visitor	Home/Personal/Education	Up to 100	Both	Development

Table 8.3: Specific Aspects of Patterns Considered Useful

When synthesising the evidence from the mapping study and the online survey, there were some consistent views about a number of patterns. For example, combining the evidence from Table 5.15 in the mapping study, Table 7.23 and Figure 7.7 in the online survey, we found that the Observer and the Composite patterns both appeared as the most favoured patterns. And also the Singleton and the Visitor patterns were considered as controversial patterns in both studies, because these two patterns appeared in both tables of useful and not useful patterns in the mapping study and the online survey. Here we synthesise the evidence for these four patterns specifically. We also apply the same method of considering general aspects to describe the patterns which have positive evidences as certified ones and the patterns which have negative or controversial evidences as uncertified ones.

8.2.2.1 Certified

1. The Observer Pattern

From the online survey, application of the Observer pattern by the respondents were mainly concentrated on small distributed system design and programming during the development stage of the software life cycle. According to the results of the mapping study, use of the Observer pattern could make design more flexible, code more understandable. But its application is restricted to expert users. If it is employed by non-expert users, the Observer pattern increases the complication of the design solution. To sum up the conclusions of both of the experience papers and the experiment papers about Observer, the application of Observer is related with whether users are familiar with it or not, even though it can make design more flexible and more understandable.

2. The Composite Pattern

The Composite pattern was mainly applied by the respondents in the small productivity / business software design and programming during the development stage of the software life cycle. From the mapping study, there were few comments other than to indicate satisfactory use. Only two experiments used

this with some students and engineers with average of 2.4 years of C++ experience. The results of one study reflected use of that the Composite pattern led to a significant increase in the time needed to complete a task, and it led participants to create overly-complicated solutions. But the replicated study with the professional software engineers did not find any negative effects to match the observation from the original study.

8.2.2.2 Uncertified

1. The Visitor Pattern

The Visitor pattern is a controversial pattern, as demonstrated in the online survey. It appears in both of the most favoured patterns list and the least favoured patterns list. And also the result of the mapping study showed that the Visitor pattern was the most commonly applied pattern in the experiment papers and the experience papers.

From the online survey, the application of the Visitor pattern by the respondents were mainly concentrated on the small home/personal/education system design and programming during the development stage of the software life cycle. Most respondents considered it as a useful pattern in both code and design level. They thought that it made a system extensible and maintainable during both the development and maintenance processes, and also that it was a good tool for decomposition and decoupling. Some respondents considered that it was good for applying in web development. Besides that it is also useful in the context of developing a language processor.

In the mapping study two experiment papers provided positive observations about the Visitor pattern, but they were concerned with how this was supported with pattern documentation. A replicated study had conflicting results, and indicated that the complexity of the Visitor pattern required more time and lower correctness. Some respondents from the online survey also had similarly negative views on it. They thought that the Visitor pattern required users to have wide experience. Otherwise it was hard to manage and maintain, especially for novices. And

also it was easy to lead to a bad structure. Sometimes it led to prohibitive time and effort being required for system design.

2. The Singleton Pattern

The Singleton pattern is also a controversial pattern in the online survey. Like the Visitor pattern, the Singleton pattern appears in both of the most favoured patterns list and the least favoured patterns list.

In the online survey, the Singleton pattern was mainly applied by the respondents in the small productivity / business software design and programming during the development stage of the software life cycle. Some respondents considered that it was massively misused and overused to get global variables according to the developer's ability. The bad coupling arising from misuse of the Singleton pattern led to poor correctness. Even one author of the GoF, Eric Gamma (Gamma *et al.*, 1995), also expressed his dislike for this pattern and would prefer to drop Singleton [1].

8.3 Threats to validity

As with any empirical study, it is necessary to consider the possible effects of any factors that could have biassed the outcomes from the study. Here we can identify three quite specific threats to validity for the mapping study that need to be considered:

- the effectiveness of the search process (internal)
- the selection and classification processes (internal)
- the completeness and the extent of the analysis of the primary studies (external)

We can also identify two threats to validity for the online survey:

[1] http://www.informit.com/articles/article.aspx?p=1404056

- the effectiveness of the responses (internal)
- the completeness of the responses (external)

and we briefly examine each of these in turn.

8.3.1 Internal Validity

8.3.1.1 The Mapping Study

For the search process, experience from conducting other studies of this form in software engineering suggests that a wide range of search engines should be employed, as no one search engine will find all of the relevant primary studies (Lesson 8 from (Brereton *et al.*, 2007)). Hence we have used a broad set of search engines, backed up by a quite thorough manual search as well as snowballing. So we are reasonably confident that this process should have identified most of the relevant studies published in the computer science and software engineering literature.

To address selection we performed this task twice. The first time, we searched for papers up to the end of 2007. To make decisions about exclusion and classification we used a basic model of 'analyst + checker', with the checker (the author' supervisor) sampling randomly, and we undertook classification independently and then resolved any differences together. Then we extended our search to the end of 2009. When we had created the extended data set, we decided to redo the inclusion/exclusion using two analysts, because it was suggested that using a model of analyst + checker could be unreliable (Turner *et al.*, 2008). We applied the two analysts model to the full data set decide which papers should be included or excluded. The inclusion/exclusion process was performed by both the author and his supervisor independently. We excluded on title firstly, then on abstract, and finally on reading the complete paper. At each stage, we resolved any differences together. Subsequent decisions about classification proved to be rather more difficult, with some aspects, such as the form of a study leading to good agreement, but with more subjective elements, such as the issue (from our set) addressed by a paper, requiring joint discussion and resolution. Again

though, since we have erred on the side of caution at each step, it seems unlikely that we have inappropriately excluded any material. We might also observe that for both selection and classification purposes, the abstracts provided were largely inadequate, often making it necessary to read parts of the paper in order to make a decision (Lesson 10 from (Brereton *et al.*, 2007)).

In terms of the lessons derived in the experience papers, we have used both established guidelines for other forms of study and also drawn from our (admittedly small) sample of selected experience papers. As such, we believe that we have addressed most of the factors relating to our chosen topic (design patterns).

8.3.1.2 The Online Survey

To collect the responses in the online survey, we used the online database in the professional survey website www.surveymonkey.com. The responses from the participants were stored in its database automatically. Before collecting these responses we sent our requests to the authors of the papers in our mapping study. But sometimes one author's name can appear in more than one paper. To avoid missing any authors, we sent the requests to the authors' different addresses. It might have led to some authors receiving the requests more than one time. Hence they could answer the questionnaire twice, creating the duplicated answers. Due to the design function issues of the SurveyMonkey website, we cannot prevent this happening or identify the duplications.

But the website provides the functions to record the identities of the respondents, such as IP address, response' time. These can be used to distinguish the duplications. And also we designed the first question in the survey form to ask the respondents provide their names and email addresses for distinguishing. We checked every response and compared with our author's list to check for any duplications. From the checking result we did not find any duplications of the responses. So through these thorough checking process we are quite confident about the effectiveness of the responses.

8.3.2 External validity

8.3.2.1 The Mapping Study

The external validity of our findings of the mapping study in terms of aggregated knowledge about the patterns is influenced by the following two factors:

- no patterns have been studied very extensively;
- The experimental studies of OO patterns mostly involved tasks that required a mix of understanding and modification, rather than the development of new systems. So the influence of patterns upon development has only been studied for a few patterns.

Another issue that perhaps should at least be noted is that most of the experiment papers are from two research groups, in Karlsruhe and Hong Kong. While this means that expertise is accumulated, it can also result in a constrained set of research strategies (such as the emphasis upon coding). While these two places do have different cultures and education systems, it does mean that the participants could behave differently to those from other cultures.

8.3.2.2 The Online Survey

During the process of collecting responses from the online survey, we found there were only few responses to the open questions. Most of the answers to the open questions were also very short and simple. Only a few of the respondents provided any details of their experiences about using patterns.

Although the open questions are not the main questions in our survey, they still provide a useful supplement for our analysis. As described in the survey analysis and the narrative synthesis, the answers from the open questions were quite useful. Perhaps the length of the survey form was longer than the respondents' expectation, or the respondents did not want to spend time typing in the open questions. Normally the respondents are reluctant to answer the questionnaire with open questions, and they are willing to answer short and simple questionnaire, especially the ones which just have the questions with options (Rayner

et al., 2004; Struwig & Stead, 2001). But if we were to design our survey form in a short and simple form, it could reduce the survey's effectiveness. Therefore, comparing between the survey effectiveness and obtaining a better response to open questions we could not sacrifice the effectiveness of the survey.

To the length of the survey form, we checked the responses including the 'excluded' responses. Some respondents only answered the "Demographic" part before exiting. But most respondents completed it, they were not influenced by the form length. So overall, the rate of completion suggests that we got the length of the survey about right.

8.4 Summary

In this chapter we synthesised the evidences from the mapping study, the online survey and the document survey about the claims by using the narrative synthesis method. We collected 12 claims and 1 factor about using design patterns previously. The evidences of the mapping study and the online survey were synthesised with them. And the results were checked to see whether there were any consistencies and differences between them.

Then we analysed the results from the general aspects and the aspects of specific patterns. From the general aspects we analysed the results to find any claims and factors that can influence software development positively or negatively. Table 8.2 summarises the results of the narrative synthesis. From the aspects of specific patterns we listed the results for each patterns, and analysed the synthesis results for some specific patterns. The Composite and Observer patterns are explicitly regarded as the useful patterns with some positive evidences. Conversely, the Visitor and Singleton patterns still have some controversial views from the users.

At the end of the synthesis process, we analysed the threats to validity for both the mapping study and the online survey from internal validity and external validity. The answers of the research questions for the narrative synthesis are presented in Chapter 9.

Chapter 9

SUMMARY AND CONCLUSIONS

9.1 Introduction

We have performed a document survey to identify claims for patterns, a mapping study, and also conducted an online survey of experienced pattern users using the list of author which was extracted from the mapping study. We have also synthesised the evidence from the mapping study, the online survey and the claim survey. This chapter reviews the overall process of our research, and summarises each study involved in the thesis. We also present the answers to the research questions posed in Chapter 1, and how the success criteria was met. In the end of this chapter some future works in research about design patterns is proposed.

9.2 Thesis Summary

This thesis consists of the claims survey, the mapping study, the online survey, and the narrative synthesis, with the mapping study and the online survey providing the primary evidence for the narrative synthesis.

To help form the research, we performed a document survey to identify claims about design patterns application. Through this survey 12 claims were extracted

from four textbooks and some papers we analysed in the mapping study. These claims were used to help formulate research questions and also in the further narrative synthesis with the results of the mapping study and the online survey.

To investigate the area of design patterns more deeply and to assess how far the claims were met, we performed a thorough and comprehensive mapping study. In this mapping study we collected 611 papers about patterns with a three-round literature search. Then these papers were classified according to the category of *Pattern theme*. In these classified papers we mainly focused on the empirical papers, and also classified them based on the categories *Form of Study* and *Pattern Issue*. In the category *Form of Study* the empirical papers were classified into *experiment* papers, *experience* papers, *case study* papers, and *survey* papers. Then we investigated on the experiment papers and the experience papers. These papers were narrowed down to two small groups of papers under a strict selection. After checking the duplication 11 experiment papers and 7 experience papers were selected for final analysis. We analysed the experiment papers and experience papers respectively, and also compared the results between them.

Through analysing the data from the mapping study, we found rather mixed evidence about the scope of usefulness for the patterns studied, and also there were little evidence in the form of any surveys. We therefore decided to perform an online survey to investigate the usefulness for the 23 design patterns. This survey was based on the information extracted from the experience papers, and sent to the authors from the papers of the mapping study and the members of three technique groups in the LinkedIn website. During the analysis process, we analysed the response data from two aspects. One aspect focused on general relations between pattern users' profile and design patterns, such as the relation of users' experiences and patterns application. The other one focused on the usefulness of specific patterns.

As the final stage of this research, we synthesised the evidence from the claims survey, the mapping study, the online survey. During this synthesis process the evidence was synthesised both from the general aspects and the specific pattern

aspects. We investigated the claims and factors which can be considered as positive or negative through the application of design patterns.

9.3 Results

As described in Chapter 1, we identified a series of research questions for each form of study. These research questions were separated into three parts. Firstly, there were four questions which were set for investigating the empirical evidences about using patterns, the forms of the empirical papers, and the claims. Secondly, only one question was posed for identifying how users think about the specific patterns during the application process and the reasons. Thirdly, at the final stage of the research three questions were employed for analysing the narrative synthesis results, and identifying which the factors of using patterns can influence the software development. Listed below are the research questions for each study and how each question was answered:

1. The mapping study:

Question-1: ***"What are the claims that are made about using design patterns?"***

In the document survey to identify claims for patterns, we identified 12 claims from four textbooks and the papers of the mapping study. These claims were the experiences being generated by experts' experiences and the previous studies. They all focus on the functions of using design patterns.

Question-2: ***"Which of the GoF patterns have been evaluated empirically?"***

This question is essentially summarised in Table 5.15. In all, 17 of the 23 patterns have been subjected to some form of formal empirical evaluation, albeit seven of them (Factory Method, Adapter, Decorator, Facade, Iterator, Strategy and Template Method) were only addressed in one study.

There was relatively little explanation in the studies that we found as to why

particular patterns were chosen. This may perhaps have been because all of these studies were conducted by software engineers who were likely to be interested in specific patterns, based upon their own experiences, rather than by cognitive scientists looking more generally at how patterns were used. However, for all but one of the papers addressing the use of OO patterns, the choice was determined by the set of applications employed to provide tasks for the participants - and the selection of these seems to have been based on availability and the need for a tractable size so that participants could understand and change the code in the available time. In some cases, the choice was of course determined by the decision to perform a replication.

The only two studies that did choose a pattern for a specific reason were FS2, where the purpose was to study the use of the Factory Method pattern for API design; and FS12 which studied the Visitor pattern. Both of these were motivated by informal experiences of the researchers, and in the case of FS12, by differences in the outcomes of previous studies.

Question-3: ***"For the GoF patterns that have been evaluated, what lessons about their use, and any consequences of their use, particularly regarding maintenance, are available from the empirical studies?"***

The mapping study provided the results generated from the experiment papers and the experience papers. But there were some contradictory results between the studies. And also there was little evidence regarding the role of patterns in maintenance.

But for the three patterns (Composite, Observer and Visitor) where there are enough studies for us to seek a more comprehensive answer to this question, Tables 5.17, 5.18 and 5.19 do offer some indication of how we can use observational studies to interpret the results from experiments - and there is also a clear warning about the risk of overly-complicated solutions if a pattern is used inappropriately.

Question-4: ***"What further research, using which forms, might be needed to address any 'gaps' in the available evidence?"***

After performing the mapping study, there was rather mixed evidence about the scope of usefulness for the patterns studied, and also little evidence about the form of survey. In particular it was clear that generic claims about the value of design patterns were inappropriate and that each pattern should be assessed separately to determine its usefulness to different groups and in different phases of software development. Therefore in order to find out more about which patterns from the set in the GoF text were perceived to be of value and hence would be likely to be used by developers and also which ones were little valued or used, as the next step we decided to perform an online survey to investigate developers' views on the specific patterns' application.

2. The online survey:

Question-5: ***"Which design patterns from the GoF, do expert pattern users consider as useful or not useful for software development and maintenance, and why?"***

This question can be divided into two questions. First of all we identified which patterns users considered as useful or not useful. Most of the respondents in this survey are experts in the areas of OO design and patterns. As shown in Tables 7.4 and 7.5, over half of them have more than ten years of experience with developing OO systems and more than 5 years of experience with using patterns. So the responses can not be influenced by the low experiences of respondents.

As demonstrated in Figure 7.8, and also combined with Figure 7.7, three patterns seem to be highly regarded with few caveats about their use (Observer, Composite and Abstract Factory), and two patterns (Memento and Flyweight) were regarded the least useful patterns.

But for the second questions the respondents did not provide enough explanations for the reasons of choosing these patterns. So we investigated them from four aspects, including system type, system size, level of abstraction, and life cycle stage. And also we found that the respondents had controversial views on using the Visitor pattern and the Singleton pattern. We analysed the qualitative

data from them.

From Tables 7.27 and 7.32, we found that most users preferred to employ design patterns in the development stage of software life cycle. Only few respondents had experiences of using patterns in the maintenance phase.

3. Evidence synthesis:

Question-6: ***"How well does the available empirical data support the claims that are made for design patterns?"***

The first objective in the final stage was to synthesis the claims with the evidences from the mapping study and the online survey. We employed the 12 claims and one factor which was extracted from the survey form for synthesis. As a result, Table 8.2 shows that four claims ("*encourage best practices*", "*vocabulary*", "*level of abstraction*", "*life cycle*") are supported positively by the evidence from the mapping study and the online survey. One claim ("*increase skills of novices*") is not supported by them. For three claims ("*reuse design & architecture*", "*improve productivity & quality*", "*property*") there are contradictory views between studies. The synthesis result also shows that developers and maintainers prefer to employ design patterns on the small size systems (factor "*system size*"). Besides that, there are no effect on other three claims ("*abstraction*", "*interface*", "*documentation*").

From the aspects of specific patterns, we synthesised the evidences for each of the patterns. Table 8.3 shows the results. The Observer and Composite patterns are considered to have positive effects on the claims. Conversely, for the Visitor and Singleton patterns there are some conflicting views between developers and maintainers.

Question-7: ***"Which patterns have been studied most widely?"***

This question is quite similar to Question-1. In this question we compared the results from Table 5.15 and Figure 7.8. The result show that the Composite, Observer, Visitor patterns are studied most widely, though there are conflicting

views about the Visitor pattern. Conversely, the patterns Memento, Builder, Mediator, Prototype, Interpreter are the least studied patterns.

Question-8: ***"How does the use of design patterns influences the process of software development?"***

From the synthesis results, we can not determine whether the application of design patterns definitely influences the development process positively. From the general aspects, the use of patterns can encourage designers' best practices and can provide a common vocabulary for designers to improve communications between designers. The use of patterns focuses on the design process, and normally in the development stage of software life cycle.

On the other side, the use of design patterns can not increase novices' skills. It should depends on design experiences. It is very difficult to say that the use of design patterns can improve programmer productivity and product quality, and can construct a product with defined properties.

9.4 Review of the Criteria for Success

In Chapter 1 we identified a series of criteria to assess how successful this study was. This section presents how these success criteria are met after performing the studies in this thesis.

1. ***Address and identify the problems and research gaps for design patterns***

In Chapters 4 and 5 we have performed a thorough mapping study. This mapping study brought us a comprehensive view on the area of design patterns. The result of the mapping study indicates rather mixed evidence about the scope of usefulness for the patterns studied. The mixed evidence that arises means that 'blind' application of patterns with any sense of the potential limitations is unwise. From the result of the mapping study we had not the necessary degree of knowledge for making evidence-based judgements about when to employ individual patterns, and also had no framework for evaluating the effectiveness of design

patterns. Besides that, we also lacked a form of survey in the collected papers of the mapping study. Therefore, the mapping study led us to conduct an online survey to investigate the values and functions of design patterns.

2. ***Exam the effectiveness of the survey***

At the beginning of the survey design, we created a protocol for conducting survey. This protocol was changed for several versions and checked by the author's supervisor. Before conducting the online survey, the design of the survey form was also checked by a pilot testing. It was reviewed by a small team of two assessors ('dry run'). After modifying the forms with the suggestions of the assessors, we sent the survey forms and the requests out.

The forms and requests were sent to 877 authors who were the paper authors of the mapping study, and three technique groups in the *LinkedIn* website. After finishing the responses collection, we collected 227 responses in total. Within these responses 206 responses were regarded as valid responses, which were well in excess of expectation.

3. ***Agreement or disagreement between the mapping study data, the survey data and the previous studies***

Most of the data from the three forms of study can be synthesised except the three claims ("*abstraction*", "*interface*", "*documentation*"). Because these claims for design patterns are entwined with a definition of what a pattern is. The consistencies or the differences between the data of the mapping study, the survey and the previous studies explain the question how the use of design patterns influences the process of software development.

4. ***Assess the effectiveness of the set of patterns provided by the GoF***

As described in the previous section, the 23 GoF design patterns were assessed in the online survey. And also the evidence about these patterns which was collected in the mapping study and the previous studies was synthesised. Table

8.3 shows the results of synthesis for each pattern. Then the Composite and Observer patterns are explicitly regarded as positive patterns during the process of development. But developers still have conflicting views on the use of the Visitor and Singleton patterns.

9.5 Directions for Further Research

After completing this thesis we have a comprehensive view about the area of design patterns, and also have the observations of how the use of design patterns influences software development process. Besides that, the results of this thesis also provide some indications about further research.

1. As analysed above, there is little evidence about their role in maintenance from the mapping study and the online survey. Table 9.1 summarises the profiles of the evidence about development and maintenance for each pattern in both studies. In the development stage all evidence from the online survey and some evidence from the mapping study concentrates on that stage. But compared with the development stage the evidence for the maintenance stage is relatively limited. Therefore, one possibility for further work could be concerned with the use of design patterns in the maintenance stage. We can conduct a similar online survey to investigate maintainers' experiences of using patterns.

2. From the results of the mapping study and the online survey, some patterns such as *Memento* and *Flyweight* are explicitly considered as not being useful patterns. In any further research we would be interested in how these patterns influence users' decision in development or maintenance, and which factors determine that users consider them as not useful.

3. Chapter 8 has noted that few answers for the open questions were collected in the survey. Therefore, in further research we would be interested in the specific comments from pattern users for some specific patterns. This could be performed by using another short survey but using more open questions. Alternatively we could conduct an observational study. The participants

Patterns	Development		Maintenance
Observer	Survey	O1, O3, O5, O7	FS4, FS7, FS8, FS10
Composite		O3	FS4, FS7, FS8, FS10
Abstract Factory		O6	FS7, O2
Facade			FS7
Proxy		O3	
Iterator		O5	
Visitor		O5	FS7, FS8, FS10, FS12
Singleton		O6	FS10
Strategy		FS3	
Adapter		FS13	
Factory Method		FS2, FS13	FS4
Template Method			FS8
Command		O3	FS4
Decorator		FS13	FS7
State		FS3, O7	FS4, O2
Bridge		O3, O5, O7	FS10
Builder			
Mediator			
Chain of Responsibility		O5	FS10
Interpreter			
Flyweight			
Prototype			
Memento			

Table 9.1: Distributions of the Studies in Development and Maintenance

of this study would be expected to choose an appropriate pattern and use it to solve a problem within a certain time. During this procedure we could observe how use of design patterns influences their design and collect think-aloud data that requires participants to verbalize their thoughts while performing this given task and generate specific data about the patterns. Then this could be analysed by using protocol analysis (Ericsson & Simon, 1993; Owen *et al.*, 2006).

4. From the responses of our survey, most of the respondents are interested in seeing the results of our survey. Therefore, in the future research we would perform a follow-on survey which is a bit more like Delphi study (Linstone & Turoff, 1975). In this follow-on survey we would mail a summary of the outcomes to the respondents which are interested in seeing anyway, and provide a short questionnaire to ask them for comments and interpretation, particularly about the maintenance implications. Then we can analyse the responses and compare with our original survey to investigate the 23 *GoF* design patterns more widely.

9.6 Concluding Remarks

Since the publication of the milestone GoF textbook , some of the 23 design patterns in this book have been widely employed in the OO development and maintenance process. The concept of a design pattern indicates that design patterns are a reuse means of documentation for successful solutions, which have been abstracted from experts' successful experiences. Theoretically design patterns can make our designs easier, and also help to create systems with properties such as flexibility and maintainability. The use of a design pattern can make a system more understandable for both designers and maintainers.

But actually design patterns are not a universal solution for more effective software development, and the application of design patterns is not so easy and perfect as the expectation. Our thesis has investigated the effectiveness of design patterns by a variety of evidence-based approaches. These studies have observed

and analysed how the functions of design patterns are influenced by some factors such as users' experiences, and observed that design patterns can not guarantee that the systems are created with defined properties. In particular, it can be seen that the successful reuse of design patterns is strongly influenced by the environment.

Appendix A

Review Protocol

The planning phase is similar to that of a systematic review (although the resulting protocol will generally be much shorter), much of the focus of a mapping study is upon the first three stages of the second phase of a review, namely:

- identification of research (searching);
- selection of primary studies (inclusion/exclusion);
- study quality assessment (bias/validity).

The data extraction stage is generally much broader than that for a systematic review, and is aimed mainly at classification and categorisation (as here).

In order to collect evidence we used digital libraries and search engines with searching terms to collect empirical patterns' papers (Kitchenham & Charters, 2007). The strategy used to construct search terms is as follows:

- Derive major terms from the questions by identifying the population, intervention and outcome;
- Use the Boolean AND to link the major term from population, intervention and outcome. We have had one round of searching and it used different search strings, as is shown in Section 4.2.

As described in (Kitchenham & Charters, 2007) we have set out our research questions for the mapping studies as above. In our protocol we adapted the form used in (von Mayrhauser & Lang, 1999). For the searching stage, the general scope of the study was identified as being:

- Population: In order to identify the academic gaps about design patterns, we collected the evidence from the published scientific literature reporting design patterns studies.
- Intervention: Empirical studies involving Patterns Application.
- Outcome of Relevance: Quantity and type of evidence relating to design patterns.
- Experimental design: Any scientific study.

Appendix B

Survey Protocol

B.1 Change Record

- Version 1: The first draft of the survey protocol
- Version 1.1: Added details in the last three sections
- Version 1.2: Modified Section 3
- Version 2: Modified Threads to validity
- Version 3: Modified 3(a)
- Version 3.1: Modified 3(d)
- Version 3.2: Modified 3(g)
- Version 4: Modified Section 5

B.2 Background

We have completed a thorough mapping study about design patterns. During this mapping study we found that most papers in recent decade focus on how to use design patterns rather than about design patterns themselves. Therefore our main research question is:

"*Which design patterns from the GoF, do expert pattern users consider as useful or not useful for software development and maintenance, and why?*"

B.3 Design

In order to explain the research questions addressed in the previous section, it is necessary for us to design a survey to collect data. These data will be applied in the future research. According to Principle of Survey Research by (Kitchenham & Pfleeger, 2002a; Pfleeger & Kitchenham, 2001), we set our survey details as following:

(a) Form of survey: Because our survey is to collect information for explain the research questions. Therefore our survey is descriptive and explanatory. We will apply a questionnaire to collect data.

(b) Data requirements: At beginning of our survey we investigate the experience papers generated from our mapping study to extract information about the all 23 patterns from GoF (Gamma *et al.*, 1995)

(c) Population: We need to find a set of participants who are experienced patterns users. They need to include both software developers and researchers.

(d) Selection of participants: There are three possible sources of experts. We can invite commercial software developers, teachers and researchers who have OO programming experiences in universities; and also numbers of engineers in the Internet expert community to take part.

(e) Sampling technique: We apply a mix of sampling. One of the sampling techniques is cluster-based sampling. And because we cannot make sure the exact population to determine a suitable sampling frame, thus we apply self-selection sampling of non-probabilistic sampling to investigate those who respond the questionnaire.

(f) Sample size: We need at least 50 responses to perform our statistical analysis.

(g) How collect the data: We mainly send our online version questionnaire via email. The response data is stored in the online database.

B.4 Data Preparation and Collection

(a) We investigate the experience papers generated from our mapping study to extract information about a certain set of design patterns. And group the related questions based on each pattern and the classified methods on GoF (Gamma *et al.*, 1995).

(b) After sending out our questionnaire we start to collect the data of responses until satisfy our confidence level. The responses will be coded according to the receive time.

(c) The data will be stored in our survey website database and also store in our local file for backup.

(d) To the questions in our questionnaire, we have to ensure that people are able and willing to answer these questions. Besides that, if we get a low response rate we can plan sending reminders to participants and individuals if necessary (Kitchenham & Pfleeger, 2002a). It can help us to assess the reason of low response.

B.5 Analysis

1. We identify the data for each pattern appeared in the mapping study and the claims. They will be combined with the survey data in the next stage for synthesis.

2. To analyse numerical data we apply the statistic forms in (Kitchenham & Pfleeger, 2003)(Kitchenham and Pfleeger, 2003). To analyse ordinal data we convert data to numerical values. To analyse nominal data we determine the proportion of responses in each category (Kitchenham & Pfleeger, 2003)(Kitchenham and Pfleeger, 2003). During the analysis process some statistic tools may be applied in our data analysis.

3. Internal validity: the effectiveness of the responses

4. External validity: the completeness of the responses

B.6 Reporting

Our target audiences are mainly composed of software engineer, students and design patterns fans. We describe the concluded data by charts and tables. A journal paper can be prepared for reporting the results.

B.7 Schedule

Time	Task
Design questionnaire	4 weeks
Release questionnaire online & collect responses	8-12 weeks
Analyse data & generate document	4 weeks

Table B.1: The Online Survey Schedule

Appendix C

Request Letter

Subject: Call for participation: On-line survey on design patterns

Dear XXX,

We are undertaking research about design patterns in software engineering. As part of a research study, we are investigating how useful or otherwise the concept of the design pattern has been demonstrated to be in practice.

The purpose of this survey is to investigate how deeply design patterns influence your experiences as a software developer, which design patterns are most frequently applied, the ones you are most reluctant to use, and the reasons for your views.

We would like to invite you to participate in this research by analysing your personal experiences and perspectives about design patterns. Your contribution will help us to analyse and assess the effectiveness of design patterns.

The data for this research will be gathered through an online questionnaire. And you will not be identified in any report that is produced using the information you have provided in this questionnaire.

We really appreciate for your generosity of sharing your experiences and perspectives about using design patterns. The survey is located at the link below and may take about 5-10 minutes to complete.

https://www.surveymonkey.com/s/LKQ3W7L

If you would be interested in seeing the results of the survey via the email and have any question about our research we will send the generated results to you and answer your question as soon as possible. Your response data will be used for the purposes of research only. We look forward to hearing from you.

Please do also copy this request to anyone else who you think might be interested in participating in this survey.

The survey will be open until Monday 2 August.

Best regards,

Cheng Zhang
PhD research student,
School of Engineering and Computing Science,
Durham University,
U.K.

Appendix D

Reminder Letter

Subject: Design Patterns Survey: Reminder

Dear XXX,

We recently invited you to take part in our survey about design patterns. This survey aims to investigate how useful or otherwise the concept of the design pattern has been demonstrated to be in practice. According to our records we have not yet received any response from you.

We understand that you are a busy person. In case the previous email was missed or you were too busy when it arrived, we are sending this reminder message. And the closing date of the survey has been extended to Tuesday 15 June. The survey is located at the link below and should take about 5-10 minutes to complete.

https://www.surveymonkey.com/s/LKQ3W7L

Do please help with our survey if you can. Clearly, the usefulness of our results depends upon receiving as many responses from experienced pattern-users as possible. Your contribution to the success of this research will be greatly appreciated.

Sincerely,

Cheng Zhang

PhD research student,
School of Engineering and Computing Science,
Durham University,
U.K.

Appendix E

Pilot Testing Questionnaire

Pilot Testing Questionnaire

This questionnaire is a pilot test for the survey of experience about design patterns. The survey is located at the link below.

https://www.surveymonkey.com/s.aspx?sm=K8oGdz3_2bJfVCIDY5ip_2bNAw_3d_3d

After completing the survey, please answer the questions below. They may take you about 10 minutes to complete. Your answers for these questions will help us to identify any errors or omissions in our survey, and help to improve the survey in terms of such aspects as design layout, clarification of questions, organisation and completeness.

1. Is the length of the survey suitable?
   Yes / No
   If No please specify whether it is too long or too short and suggest any possible ways of addressing this.

2. Do the item numbers make the structure of the survey clear to the respondents?
   Yes / No

3. Is the layout of the survey clear?
   Yes / No
   If No please specify which parts are unclear.

4. Is the organisation of the questions reasonable?
   Yes / No

5. Are the questions in the survey easy to understand?
   Yes / No
   If No please identify those that are unclear.

6. Are the questions of the survey comprehensive?
   Yes / No
   If No please specify what you think is lacking.

7. Are the response choices mutually exclusive?
   Yes / No
   If No please specify them.

8. Are the response choices exhaustive?
   Yes / No
   If No please specify what you think should be added.

9. Are the instructions for completing each part of the survey clearly written?
   Yes / No
   If No please specify which questions are not clearly expressed.

Appendix F

Experiment Papers

No.	*Publication details*
FS1	Chung E, Hong J, Lin J, Prabaker M, Landay J & Liu A (2004). Development and evaluation of emerging design patterns for ubiquitous computing", In *Proceedings of DIS'04: Conference on Designing Interactive Systems*, ACM Press, 233 - 242.
FS2	Ellis B, Stylos J & Myers B (2007). The Factory Pattern in API Design: A Usability Evaluation, In *Proceedings of the 29th International Conference on Software Engineering: ICSE'07*, IEEE Computer Society Press, 302 - 311.
FS3	Ng TH, Cheung SC, Chan WK & Yu YT (2006). Toward effective deployment of design patterns for software extension: a case study, In *Proceedings of the International Workshop on Software Quality: WoSQ'06*, ACM Press, 51 - 56.
FS4	Ng TH, Cheung SC, Chan WK & Yu YT (2007). Do Maintainers Utilize Deployed Design Patterns Effectively?, In *Proceedings of the 29th International Conference on Software Engineering: ICSE'07*, IEEE Computer Society Press, 168 - 177.
FS5	Prechelt L & Unger B (1998). A Series of Controlled Experiments on Design Patterns: Methodology and Results, In *Proceedings of Softwaretechnik'98*, Paderborn.
FS6	Prechelt L, Unger B & Philippsen M (1997). Documenting design patterns in code eases program maintenance, In *Proceedings ICSE Workshop on Process Modelling and Empirical Studies of Software Evolution*, IEEE Computer Society Press, 72 - 76. [Removed from study as data duplicated in FS8]

Continued on next page

– continued from previous page

No.	*Publication details*
FS7	Prechelt L, Unger B, Tichy WF, Brössler P & Votta LG (2001). A Controlled Experiment in Maintenance Comparing Design Patterns to Simpler Solutions, *IEEE Transactions on Software Engineering*, 27(12), 1134 - 1144.
FS8	Prechelt L, Unger-Lamprecht B, Philippsen M & Tichy WF (2002). Two Controlled Experiments Assessing the Usefulness of Design Pattern Documentation in Program Maintenance, *IEEE Transactions on Software Engineering*, 28(6), 595 - 606.
FS9	Torchiano M (2002). Documenting pattern use in Java programs, In *Proceedings of International Conference on Software Maintenance (ICSM'02)*, 230 - 233.
FS10	Unger, B. & Tichy, W. (2000). Do Design Patterns improve Communication? An Experiment with Pair Design. In *Proceedings International Workshop on Empirical Studies of Software Maintenance*, 1 - 5.
FS11	Vokáč, M., Tichy, W.F., Sjøberg, D.I.K., Arisolm, E. & Aldrin, M. (2004). A controlled experiment comparing the maintainability of programs designed with and without design patterns - a replication in a real programming environment. *Empirical Software Engineering*, 9, 149 - 195.
FS12	Jeanmart, S., Guéhéneuc, Y.G., Sahraoui, H. & Habra, N. (2009). Impact of the visitor pattern on program comprehension and maintenance. In *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, ESEM '09, 69-78, IEEE Computer Society, Washington, DC, USA.
FS13	Abdul Jalil, M. & Noah, S.A.M. (2007). The difficulties of using design patterns among novices: An exploratory study. In *Proceedings of the The 2007 International Conference Computational Science and its Applications*, ICCSA '07, 97 - 103, IEEE Computer Society, Washington, DC, USA.

Appendix G

Experience Papers

No.	*Publication details*
O1	Schmidt, D. (1995). Using Design Patterns to develop reusable Object-Oriented Communication Software. *Communications of the ACM*, 38, 65 - 74.
O2	Yuanhong, W., Hong, M. & Weizhong, S. (1997). Experience report: Using design patterns in the development of jb system. In *Proceedings of Technology of Object Oriented Languages and Systems: TOOLS 24*, 159 - 165, IEEE Computer Society Press.
O3	Wendorff, P. (2001). Assessment of design patterns during software reengineering: Lessons learned from a large commercial project. In *Proceedings of 5th European Conference on Software Maintenance and Reengineering (CSMR'01)*, 77-84, IEEE Computer Society Press.
O4	Masuda, G., Sakamoto, N. & Ushijima, K. (1998). Applying design patterns to decision tree learning system. In *Proceedings of 6th ACM SIGSOFT International Symposium on Foundations of Software Engineering SIGSOFT'98/FSE-6*, 111 - 120, ACM Press.
O5	Wydaeghe, B., Verschaeve, K., Michiels, B., Damme, B.V., Archens, E. & Jonckers, V. (1998). Building an omt-editor using design patterns: An experience report. In *Proceedings of Conference on Technology of Object-Oriented Languages*, TOOLS.
O6	Cinnéeide, M.Ó. & Fagan, P. (2006). Design patterns: The devils in the detail. In *Proceedings of 2006 Conference on Pattern Languages of Programs, PLoP'06*.
O7	Chatzigeorgiou, A., Tsantalis, N. & Deligiannis, I. (2008). An empirical study on students' ability to comprehend design patterns. *Computers & Education*, 51, 1007-1016.

Appendix H

The Online Questionnaire

I. Introduction

As part of a research study, we are investigating how useful or otherwise the concept of the design pattern has been demonstrated to be in practice.

The purpose of this survey is to therefore investigate how deeply design patterns influence software developers' experiences, which design patterns are most frequently applied, the ones they are most reluctant to use, and the reasons for their views.

We begin by asking about your experiences with using patterns, and your views about those that are familiar to you. We then ask you to provide us with a little more information about the ones that you consider most and least useful.

We really appreciate for your generosity of sharing your experiences and perspectives about using design patterns. Please press the "Next" button below to start this survey, and it may take you about 5-10 minutes to complete.

II. About you

1. Please note you will not be identified in any report that is produced using the information you have provided in this questionnaire.
   Your name:
   Email address:

2. Which of the following best describes your primary role during software development?

   a) Commercial software developer

   b) Software researcher

   c) Software teacher

   d) Student of computing

3. Highest degree you have earned?

   a) Associate degree

   b) Bachelors degree

   c) Masters

   d) Ph.D. or equivalent

   e) Other

4. How many years of experience do you have with Object-Oriented development?

   a) Less than 3 years

   b) 3 to 5 years

   c) 6 to 10 years

   d) 11 to 15 years

e) Over 15 years

5. How many years of experience do you have with working with design patterns?

   a) Less than 3 years

   b) 3 to 5 years

   c) 6 to 10 years

   d) 11 to 15 years

6. Have you written any patterns (or rewritten any existing patterns)?

   a) Yes

   b) No

7. In this section you are asked to provide us with your assessment of the usefulness of each pattern in the book "Design Patterns: Elements of Reusable Object-Oriented Software" by Gamma et al., based upon your experiences with using that pattern.
   (For each pattern identify as: *Very Useful*; *Useful*; *Not Very Useful*; *Not at all Useful*; *Little or no Experience of using this pattern*)

III. Design patterns evaluation (Pattern considered most useful)

8. From the same list of patterns, we would like to know your views and experiences of up to three patterns that you have found MOST useful. On the list below, please identify the FIRST pattern that you have found to be most useful. (If you have fewer than three patterns, when you have completed your responses, use the Not Applicable option in the list of patterns

below and use the 'Next' button to proceed on to the next set of questions.)

- (1) Abstract Factory
- (2) Adapter
- (3) Bridge
- (4) Builder
- (5) Chain of Responsibility
- (6) Command
- (7) Composite
- (8) Decorator
- (9) Facade
- (10) Factory Method
- (11) Flyweight
- (12) Interpreter
- (13) Iterator
- (14) Mediator
- (15) Memento
- (16) Observer
- (17) Prototype
- (18) Proxy
- (19) Singleton
- (20) State
- (21) Strategy
- (22) Template Method
- (23) Visitor
- Not Applicable

9. What type(s) of software have you developed or maintained with this pattern? (You can check more than one.)

   a) Productivity/Business Software

   b) Graphic Design and Multimedia

   c) Home/Personal/Education

   d) Distributed systems (such as web-based application)

   e) System Software (e.g. Operating system, Middleware)

   Other (please specify):

10. What was the size of the system? KLOC

    a) up to 100;

    b) 100-250

    c) 250 to 500

    d) above 500

    Number of classes:

11. What was the level of abstraction involved in using this design pattern?

    a) Design

    b) Code

    c) Both

12. In what stage(s) in the life-cycle of the system(s) did you work with this design pattern?

    a) Development

    b) Maintenance

13. Please describe the experiences that form your reasons for liking this design pattern, the characteristic of the pattern that you found most useful, and why.

Questions 8-13 are repeated twice more if the responder continues to input.

IV. Design patterns evaluation (Pattern not considered useful)

And then we use a similar structure of the previous part to identify patterns that were not found to be useful.

V. Other

The final questions ask whether the user has used any other patterns and offer a free-text opportunity to make other observations about design patterns.

Appendix I

Summary of Overall Pattern Usefulness in Three Groups

			Overall Profile of Usefulness					Total
			Little or No Experience(LE)	Not at all Useful(NU)	Not Very Useful(NVU)	Useful(U)	Very Useful(VU)	
Abstract Factory	Author	Count	12	1	8	53	51	125
		% within Group	9.6%	0.8%	6.4%	42.4%	40.8%	100.0%
	Merged	Count	4	1	2	21	23	51
		% within Group	7.8%	2.0%	3.9%	41.2%	45.1%	100.0%
Builder	Author	Count	34	1	24	49	17	125
		% within Group	27.2%	0.8%	19.2%	39.2%	13.6%	100.0%
	Merged	Count	11	1	7	23	9	51
		% within Group	21.6%	2.0%	13.7%	45.1%	17.6%	100.0%
Factory Method	Author	Count	10	2	7	46	60	125
		% within Group	8.0%	1.6%	5.6%	36.8%	48.0%	100.0%
	Merged	Count	4	0	3	17	27	51
		% within Group	7.8%	0.0%	5.9%	33.3%	52.9%	100.0%
Prototype	Author	Count	32	2	30	46	15	125
		% within Group	25.6%	1.6%	24.0%	36.8%	12.0%	100.0%
	Merged	Count	20	2	9	12	8	51
		% within Group	39.2%	3.9%	17.6%	23.5%	15.7%	100.0%
Singleton	Author	Count	6	5	19	31	64	125
		% within Group	4.8%	4.0%	15.2%	24.8%	51.2%	100.0%
	Merged	Count	1	5	8	12	25	51
		% within Group	2.0%	9.8%	15.7%	23.5%	49.0%	100.0%

Continued on next page

Table I.1 – continued from previous page

			Overall Profile of Usefulness					Total
			Little or No Experience(LE)	Not at all Useful(NU)	Not Very Useful(NVU)	Useful(U)	Very Useful(VU)	
Adapter	Author	Count	10	2	3	46	64	125
		% within Group	8.0%	1.6%	2.4%	36.8%	51.2%	100.0%
	Merged	Count	4	1	1	20	25	51
		% within Group	7.8%	2.0%	2.0%	39.2%	49.0%	100.0%
Bridge	Author	Count	27	3	7	60	28	125
		% within Group	21.6%	2.4%	5.6%	48.0%	22.4%	100.0%
	Merged	Count	18	1	5	21	6	51
		% within Group	35.3%	2.0%	9.8%	41.2%	11.8%	100.0%
Composite	Author	Count	5	2	4	29	85	125
		% within Group	4.0%	1.6%	3.2%	23.2%	68.0%	100.0%
	Merged	Count	8	1	5	12	25	51
		% within Group	15.7%	2.0%	9.8%	23.5%	49.0%	100.0%
Decorator	Author	Count	14	3	14	46	48	125
		% within Group	11.2%	2.4%	11.2%	36.8%	38.4%	100.0%
	Merged	Count	11	1	5	17	17	51
		% within Group	21.6%	2.0%	9.8%	33.3%	33.3%	100.0%
Façade	Author	Count	14	1	12	44	54	125
		% within Group	11.2%	0.8%	9.6%	35.2%	43.2%	100.0%
	Merged	Count	4	1	3	19	24	51
		% within Group	7.8%	2.0%	5.9%	37.3%	47.1%	100.0%

Continued on next page

Table I.1 – continued from previous page

			Overall Profile of Usefulness					Total
			Little or No Experience(LE)	Not at all Useful(NU)	Not Very Useful(NVU)	Useful(U)	Very Useful(VU)	
Flyweight	Author	Count	43	8	25	38	11	125
		% within Group	34.4%	6.4%	20.0%	30.4%	8.8%	100.0%
	Merged	Count	19	1	9	19	3	51
		% within Group	37.3%	2.0%	17.6%	37.3%	5.9%	100.0%
Proxy	Author	Count	15	1	4	45	60	125
		% within Group	12.0%	0.8%	3.2%	36.0%	48.0%	100.0%
	Merged	Count	6	0	4	16	25	51
		% within Group	11.8%	0.0%	7.8%	31.4%	49.0%	100.0%
Command	Author	Count	17	2	7	53	46	125
		% within Group	13.6%	1.6%	5.6%	42.4%	36.8%	100.0%
	Merged	Count	11	0	4	18	18	51
		% within Group	21.6%	0.0%	7.8%	35.3%	35.3%	100.0%
Interpreter	Author	Count	41	10	22	33	19	125
		% within Group	32.8%	8.0%	17.6%	26.4%	15.2%	100.0%
	Merged	Count	19	2	9	18	3	51
		% within Group	37.3%	3.9%	17.6%	35.3%	5.9%	100.0%
Iterator	Author	Count	11	1	9	40	64	125
		% within Group	8.8%	0.8%	7.2%	32.0%	51.2%	100.0%
	Merged	Count	5	0	2	15	29	51
		% within Group	9.8%	0.0%	3.9%	29.4%	56.9%	100.0%

Continued on next page

Table I.1 – continued from previous page

			Overall Profile of Usefulness					Total
			Little or No Experience(LE)	Not at all Useful(NU)	Not Very Useful(NVU)	Useful(U)	Very Useful(VU)	
Mediator	Author	Count	29	1	24	45	26	125
		% within Group	23.2%	0.8%	19.2%	36.0%	20.8%	100.0%
	Merged	Count	21	1	7	14	8	51
		% within Group	41.2%	2.0%	13.7%	27.5%	15.7%	100.0%
Memento	Author	Count	47	9	24	37	8	125
		% within Group	37.6%	7.2%	19.2%	29.6%	6.4%	100.0%
	Merged	Count	23	3	8	11	6	51
		% within Group	45.1%	5.9%	15.7%	21.6%	11.8%	100.0%
Observer	Author	Count	5	2	2	28	88	125
		% within Group	4.0%	1.6%	1.6%	22.4%	70.4%	100.0%
	Merged	Count	4	0	5	15	27	51
		% within Group	7.8%	0.0%	9.8%	29.4%	52.9%	100.0%
State	Author	Count	18	2	19	43	43	125
		% within Group	14.4%	1.6%	15.2%	34.4%	34.4%	100.0%
	Merged	Count	14	1	8	17	11	51
		% within Group	27.5%	2.0%	15.7%	33.3%	21.6%	100.0%
Strategy	Author	Count	16	1	14	43	51	125
		% within Group	12.8%	0.8%	11.2%	34.4%	40.8%	100.0%
	Merged	Count	8	1	1	17	24	51
		% within Group	15.7%	2.0%	2.0%	33.3%	47.1%	100.0%

Continued on next page

Table I.1 – continued from previous page

			Overall Profile of Usefulness					Total
			Little or No Experience(LE)	Not at all Useful(NU)	Not Very Useful(NVU)	Useful(U)	Very Useful(VU)	
Template Method	Author	Count	19	1	10	38	57	125
		% within Group	15.2%	0.8%	8.0%	30.4%	45.6%	100.0%
	Merged	Count	11	3	5	14	18	51
		% within Group	21.6%	5.9%	9.8%	27.5%	35.3%	100.0%
Visitor	Author	Count	14	3	12	47	49	125
		% within Group	11.2%	2.4%	9.6%	37.6%	39.2%	100.0%
	Merged	Count	10	4	5	12	20	51
		% within Group	19.6%	7.8%	9.8%	23.5%	39.2%	100.0%
Chain of Responsibility	Author	Count	26	4	17	54	24	125
		% within Group	20.8%	3.2%	13.6%	43.2%	19.2%	100.0%
	Merged	Count	11	1	4	24	11	51
		% within Group	21.6%	2.0%	7.8%	47.1%	21.6%	100.0%

References

Abdul Jalil, M. & Noah, S.A.M. (2007). The difficulties of using design patterns among novices: An exploratory study. In *Proceedings of the The 2007 International Conference Computational Science and its Applications*, ICCSA '07, 97–103, IEEE Computer Society, Washington, DC, USA.

Alexander, C. (1964). *Notes on the Synthesis of Form (Harvard Paperbacks)*. Harvard University Press.

Alexander, C. (1975). *The Oregon Experiment (Center for Environmental Structure Series)*. Oxford University Press, USA.

Alexander, C. (1979). *The Timeless Way of Building*. Oxford University Press.

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I. & Angel, S. (1977). *A Pattern Language: Towns, Buildings, Construction*. Oxford University Press.

Anderson, B. (1992). Towards an architecture handbook. In *Addendum to the proceedings on Object-oriented programming systems, languages, and applications (Addendum)*, OOPSLA '92, 167–168, ACM, New York, NY, USA.

Beck, K. (1988). Using a pattern language for programming. In *Addendum to the Proceedings of OOPSLA '87*, vol. 23, 16, ACM SIGPLAN Notices.

Beck, K., Crocker, R., Meszaros, G., Vlissides, J., Coplien, J.O., Dominick, L. & Paulisch, F. (1996). Industrial experience with design patterns. In *ICSE '96: Proceedings of the 18th international conference on Software engineering*, 103–114, IEEE Computer Society, Washington, DC, USA.

Bernstein, R. & Bernstein, S. (1999). *Schaum's Outline of Elements of Statistics II: Inferential Statistics*. McGraw-Hill.

Bernstein, S. & Bernstein, R. (1998). *Schaum's Outline of Elements of Statistics I: Descriptive Statistics and Probability*. McGraw-Hill, 1st edn.

Bieman, J.M., Straw, G., Wang, H., Munger, P.W. & Alexander, R.T. (2003). Design patterns and change proneness: an examination of five evolving systems. In *Software Metrics Symposium, 2003. Proceedings. Ninth International*, 40–49.

Biemer, P. & Lyberg, L. (2003). *Introduction to Survey Quality (Wiley Series in Survey Methodology)*. Wiley-Interscience.

Bourque, L. & Fielder, E. (1995). *How to Conduct Self-Administered and Mail Surveys (Survey Kit, Vol 3)*. Sage Publications, Inc.

Brace, I. (2004). *Questionnaire Design: How to Plan, Structure and Write Survey Material for Effective Market Research with CDROM (Market Research in Practice Series)*. Kogan Page.

Brereton, P., Kitchenham, B.A., Budgen, D., Turner, M. & Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software*, **80**, 571–583.

Brereton, P., Kitchenham, B. & Budgen, D. (2008). Using a protocol template for case study planning. In *Proceedings of EASE 2008*, BCS-eWiC.

Bronson, G. (2009). *C++ for Engineers and Scientists*. Course Technology.

Brooks, A. (1997). Meta analysis a silver bullet for meta-analysts. *Empirical Softw. Engg.*, **2**, 333–338.

Brue, G. (2005). *Six Sigma for Managers: 24 Lessons to Understand and Apply Six Sigma Principles in Any Organization (The McGraw-Hill Professional Education Series)*. McGraw-Hill, 1st edn.

Budgen, D. & Zhang, C. (2009). Preliminary reporting guidelines for experience papers. In *13th International Conference on Evaluation and Assessment in Software Engineering (EASE)*, 1–10, BCS eWiC.

Budinsky, F.J., Finnie, M.A., Vlissides, J.M. & YU, P.S. (1996). Automatic code generation from design patterns. *IBM Syst. J.*, **35**, 151–171.

Buschmann, F., Meunier, R., Rohnert, H., Sommerlad, P. & Stal, M. (1996). *Pattern-Oriented Software Architecture Volume 1: A System of Patterns*. Wiley.

Buschmann, F., Henney, K. & Schmidt, D. (2007). *Pattern-Oriented Software Architecture Volume 4: A Pattern Language for Distributed Computing*. Wiley.

Buschmann, H.K., Frank & Schmidt, D. (2007). *Pattern-Oriented Software Architecture Volume 5: On Patterns and Pattern Languages*. Wiley.

Chatzigeorgiou, A., Tsantalis, N. & Deligiannis, I. (2008). An empirical study on students' ability to comprehend design patterns. *Comput. Educ.*, **51**, 1007–1016.

Chung, E., Hong, J., Prabaker, M., Landay, J. & Liu, A. (2004). Development and evaluation of emerging design patterns for ubiquitous computing. In *Proceedings of DIS'04: Conference on Designing Interactive Systems*, 233–242, ACM Press.

Cinnéide, M.Ó. & Fagan, P. (2006). Design patterns: The devils in the detail. In *Proceedings of 2006 Conference on Pattern Languages of Programs, PLoP'06*.

Ciolkowski, M. (2009). What do we know about perspective-based reading? an approach for quantitative aggregation in software engineering. In *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, ESEM '09, 133–144, IEEE Computer Society, Washington, DC, USA.

Ciolkowski, M. & Münch, J. (2005). Accumulation and presentation of empirical evidence: problems and challenges. In *Proceedings of the 2005 workshop on Realising evidence-based software engineering*, REBSE '05, 1–3, ACM, New York, NY, USA.

Cooper, H. (2009). *The Handbook of Research Synthesis and Meta-Analysis*. Russell Sage Foundation Publications, 2nd edn.

Coplien, J. (1991). *Advanced C++ Programming Styles and Idioms*. Addison-Wesley.

Coplien, J.O. & Schmidt, D.C., eds. (1995). *Pattern languages of program design*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA.

Cox, D.R., Commenges, D., Davison, A.C., Solomon, P.J. & Wilson, S.R. (2003). The Oxford Dictionary of Statistical Terms. In Y. Dodge, ed., *The Oxford Dictionary of Statistical Terms*, Oxford University Press.

Cruzes, D.S. & Dybå, T. (2010). Synthesizing evidence in software engineering research. In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, ESEM '10, 1:1–1:10, ACM, New York, NY, USA.

Dasiewicz, P. (2005). Design patterns and object-oriented software testing. In *Electrical and Computer Engineering, 2005. Canadian Conference on*, 904–907.

Davis, A., Dieste, O., Hickey, A., Juristo, N. & Moreno, A.M. (2006). Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review. In *RE '06: Proceedings of the 14th IEEE International Requirements Engineering Conference (RE'06)*, 176–185, IEEE Computer Society.

Decyk, V. & Gardner, H. (2007). A factory pattern in fortran 95. In *ICCS '07: Proceedings of the 7th international conference on Computational Science, Part I*, 583–590, Springer-Verlag.

Diamond, S.S. (2000). *Reference guide on survey research,*, 229–276. Federal Judicial Center, Washington, D.C., 2nd edn.

Dillman, D. (1999). *Mail and Internet Surveys: The Tailored Design Method*. Wiley.

Dixon-Woods, M., Agarwal, S., Jones, D., Young, B. & Sutton, A. (2005). Synthesising qualitative and quantitative evidence: a review of possible methods. *J Health Serv Res Policy*, **10**, 45–53.

Dong, J., Peng, T. & Qiu, Z. (2007). Commutability of design pattern instantiation and integration. In *TASE'07*, 283–292.

Dybå, T., Kitchenham, B. & Jørgensen, M. (2005). Evidence-based software engineering for practitioners. *IEEE Software*, **22**, 58–65.

Ellis, B., Stylos, J. & Myers, B. (2007). The factory pattern in API design: A usability evaluation. In *Proceeding of 29th International Conference on Software Engineering (ICSE 2007)*, 302–311, IEEE Computer Society Press.

Ericsson, A. & Simon, H. (1993). *Protocol Analysis - Rev'd Edition: Verbal Reports as Data*. The MIT Press.

Fink, A.G. (2002). *The Survey Handbook 2nd edition*. the Survey Kit, Sage Publications, Inc.

Florijn, M.M., Gert & van Winsen, P. (1997). Tool support for object-oriented patterns. In M. Ak?it & S. Matsuoka, eds., *ECOOP'97 - Object-Oriented Programming*, vol. 1241, 472–495, Springer Berlin / Heidelberg.

Forbig, P. & Lammel, R. (2000). Programming with patterns. In *Technology of Object-Oriented Languages and Systems, 2000. TOOLS 34. Proceedings. 34th International Conference on*, 159–170.

Freeman, E., Freeman, E., Kathy, S. & Bert, B. (2004). *Head First Design Patterns*. O'Reilly Media.

Gamma, E. (1991). *Object-Oriented Software Development based on ET++*. Ph.D. thesis, University of Zurich, Institut für Informatik. (in German), also available through Springer-Verlag, Berlin, 1992.

Gamma, E., Helm, R., Johnson, R.E. & Vlissides, J.M. (1993). Design patterns: Abstraction and reuse of object-oriented design. In *Proceedings of the 7th European Conference on Object-Oriented Programming*, ECOOP '93, 406–431, Springer-Verlag, London, UK.

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

Geisser, S. & Johnson, W. (2006). *Modes of Parametric Statistical Inference (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 1st edn.

Glass, R., Vessey, I. & Ramesh, V. (2002). Research in software engineering: an analysis of the literature. *Information and Software Technology*, **44**, 491–506.

Glass, R., Ramesh, V. & Vessey, I. (2004). An analysis of research in computing disciplines. *Commun. ACM*, **47**, 89–94.

Gravetter, F. & Wallnau, L. (2007). *Essentials of Statistics for the Behavioral Science*. Wadsworth Publishing, 6th edn.

Hayes, W. (1999). Research synthesis in software engineering: A case for meta-analysis. In *Proceedings of the 6th International Symposium on Software Metrics*, 143–, IEEE Computer Society, Washington, DC, USA.

Heer, J. & Agrawala, M. (2006). Software design patterns for information visualization. *Visualization and Computer Graphics, IEEE Transactions on*, **12**, 853–860.

Höst, M. & Runeson, P. (2007). Checklists for software engineering case study research. In *1st Int. Symposium on Empirical Software Engineering and Measurement (ESEM)*, 479–481, IEEE Computer Society Press.

Howell, D. (1998). *Fundamental Statistics for the Behavioral Sciences (with CD-ROM)*. Duxbury Resource Center, 4th edn.

Iarossi, G. (2006). *The Power of Survey Design: A User's Guide for Managing Surveys, Interpreting Results, and Influencing Respondents*. World Bank Publications.

Jeanmart, S., Gueheneuc, Y.G., Sahraoui, H. & Habra, N. (2009). Impact of the visitor pattern on program comprehension and maintenance. In *Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, ESEM '09, 69–78, IEEE Computer Society, Washington, DC, USA.

Jeffery, D.R. & Votta, L.G. (1999). Empirical software engineering. *Software Engineering, IEEE Transactions on*, **25**, 435–437.

Jones, C. (2009). *Software Engineering Best Practices: Lessons from Successful Projects in the Top Companies*. McGraw-Hill Osborne Media, 1st edn.

Jorgensen, D.L. (1989). *Participant Observation: A Methodology for Human Studies (Applied Social Research Methods)*. Sage Publications, Inc.

Kambayashi, Y. & Ohki, M. (2003). Extracting the software elements and design patterns from the software field. In *Proc. of ICEIS03*.

Kennedy, G. (2004). Design patterns in information system development. In *Proc. 5th Australasian Workshop on Software and System*.

Kerth, N.L. & Cunningham, W. (1997). Using patterns to improve our architectural vision. *IEEE Softw.*, **14**, 53–59.

Khomh, F. & Guéhéneuc, Y.G. (2007). Perception and reality: What are design patterns good for? In B. Fernando, C. Calero, Y.G. Guéhéneuc, C. Lange, M. Lanza & H. Sahraoui, eds., *Proceedings of the 11th ECOOP workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE)*, Springer-Verlag. Note: 7 pages.

Kim, D. & Khawand, C. (2007). An approach to precisely specifying the problem domain of design patterns. *J. Vis. Lang. Comput.*, **18**, 560–591.

Kircher, M. & Jain, P. (2004). *Pattern-Oriented Software Architecture Volume 3: Patterns for Resource Management*. Wiley.

Kirk, R. (1994). *Experimental Design: Procedures for Behavioral Sciences (Psychology)*. Wadsworth Publishing, 3rd edn.

Kitchenham, B. & Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Tech. rep.

Kitchenham, B. & Pfleeger, S.L. (2002a). Principles of survey research part 2: designing a survey. *SIGSOFT Softw. Eng. Notes*, **27**, 18–20.

Kitchenham, B. & Pfleeger, S.L. (2002b). Principles of survey research part 3: constructing a survey instrument. *SIGSOFT Softw. Eng. Notes*, **27**, 20–24.

Kitchenham, B. & Pfleeger, S.L. (2002c). Principles of survey research part 4: questionnaire evaluation. *SIGSOFT Softw. Eng. Notes*, **27**, 20–23.

Kitchenham, B. & Pfleeger, S.L. (2002d). Principles of survey research part 5: populations and samples. *SIGSOFT Softw. Eng. Notes*, **27**, 17–20.

Kitchenham, B. & Pfleeger, S.L. (2003). Principles of survey research part 6: data analysis. *SIGSOFT Softw. Eng. Notes*, **28**, 24–27.

Kitchenham, B.A., Budgen, D. & Brereton, O.P. (2010). Using mapping studies as the basis for further research - a participant-observer case study. *Information and Software Technology*, **In Press, Corrected Proof**, –.

Lea, D. (1994). Christopher alexander: an introduction for object-oriented designers. *SIGSOFT Softw. Eng. Notes*, **19**, 39–46.

Lee, H., Youn, H. & Lee, E. (2007). Automatic detection of design pattern for reverse engineering. In *5th ACIS International Conference on Software Engineering Research, Management & Applications (SERA 2007)*, 577–583, IEEE.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of Psychology*, **140**.

Lindsay, R.M. & Ehrenberg, A.S.C. (1993). The design of replicated studies. *The American Statistician*, **47**, 12.

Linstone, H.A. & Turoff, M. (1975). *The Delphi method : techniques and applications*. Addison-Wesley Pub. Co., Advanced Book Program, Reading, Mass. :.

Mak, J., Choy, C. & Lun, D. (2004). Precise modeling of design patterns in uml. In *Proceedings of the 26th International Conference on Software Engineering*, 252–261, IEEE Computer Society.

Mann, P.S. (2006). *Introductory Statistics*. Wiley, 6th edn.

Mapelsden, D., Hosking, J. & Grundy, J. (2002). Design pattern modelling and instantiation using dpml. In *CRPIT '02: Proceedings of the Fortieth International Conference on Tools Pacific*, 3–11, Australian Computer Society, Inc.

Martin, R. (1995). *Discovering patterns in existing applications*, 365–393. ACM Press/Addison-Wesley Publishing Co.

Masuda, G., Sakamoto, N. & Ushijima, K. (1998). Applying design patterns to decision tree learning system. In *Proceedings of 6th ACM SIGSOFT International Symposium on Foundations of Software Engineering SIGSOFT'98/FSE-6*, 111–120, ACM Press.

Meister, J., Reussner, R. & Rohde, M. (2004). Applying patterns to develop a product line architecture for statistical analysis software. In *Software Architecture, 2004. WICSA 2004. Proceedings. Fourth Working IEEE/IFIP Conference on*, 291–294.

Meszaros, G. & Doble, J. (1997). *A pattern language for pattern writing*, 529–574. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Mili Hafedh, Y.S.A.E., Mili Ali (2002). *Reuse-Based Software Engineering: Techniques, Organizations, and Controls*. Wiley-Interscience.

Miller, J. (2000). Applying meta-analytical procedures to software engineering experiments. *J. Syst. Softw.*, **54**, 29–39.

Naps, T. & Nance, D. (1995). *Introduction to Computer Science: Programming, Problem Solving and Data Structures, Alternate Edition*. West Publishing Company.

Ng, T.H. & Cheung, S.C. (2005). Enhancing class commutability in the deployment of design patterns. *Information and Software Technology*, **47**, 797–804.

Ng, T.H., Cheung, S.C., Chan, W.K. & Yu, Y.T. (2006). Toward effective deployment of design patterns for software extension: a case study. In *Proceedings of International Workshop on Software Quality WoSQ'06*, 51–56, ACM Press.

Ng, T.H., Cheung, S.C., Chan, W.K. & Yu, Y.T. (2007). Do maintainers utilize deployed design patterns effectively? In *Proceeding of 29th International Conference on Software Engineering (ICSE 2007)*, 168–177, IEEE Computer Society Press.

Noda, N. & Kishi, T. (2001). Implementing design patterns using advanced separation of concerns. In *OOPSLA 2001 workshop on Advanced Separation of Concerns in Object-Oriented Systems*.

Oates, B. (2005). *Researching Information Systems and Computing*. Sage Publications Ltd.

O'Docherty, M. (2005). *Object-Oriented Analysis and Design: Understanding System Development with UML 2.0*. Wiley.

Ogawa, R. & Malen, B. (1991). Towards rigor in reviews of multivocal literatures: Applying the exploratory case study method. *Review of Educational Research*, **61**, 265–286.

Onions, C.T., Friedrichsen, G.W.S. & Burchfield, R.W., eds. (1966). *The Oxford Dictionary of English Etymology*. Oxford University Press, USA.

Owen, S., Brereton, P. & Budgen, D. (2006). Protocol analysis: a neglected practice. *Commun. ACM*, **49**, 117–122, 1113039.

Perry, D., Porter, A. & Votta, L. (2000). Empirical studies of software engineering: a roadmap. In *ICSE '00: Proceedings of the Conference on The Future of Software Engineering*, 345–355, ACM, Limerick, Ireland.

Petticrew, M. & Roberts, H. (2005). *Systematic Reviews in the Social Sciences: A Practical Guide*. Wiley-Blackwell.

Pfleeger, S.L. & Kitchenham, B. (2001). Principles of survey research part 1: turning lemons into lemonade. *SIGSOFT Softw. Eng. Notes*, **26**, 16–18.

Popay, J., Roberts, H., Sowden, A., Petticrew, M., Arai, L., Rodgers, M., Britten, N., Roen, K. & Duffy, S. (2006). Guidance on the conduct of narrative synthesis in systematic reviews: a product of the esrc methods programme (version i). Tech. rep., Lancaster: Institute of Health Research.

Prechelt, L., Unger, B., Tichy, W.F., Brossler, P. & Votta, L.G. (2001). A controlled experiment in maintenance comparing design patterns to simpler solutions. *IEEE Transactions on Software Engineering*, **27**, 1134–1144.

Prechelt, L., Unger-Lamprecht, B., Philippsen, M. & Tichy, W.F. (2002). Two controlled experiments assessing the usefulness of design pattern documentation in program maintenance. *IEEE Transactions on Software Engineering*, **28**, 595–606.

Pree, W. (1995). *Design Patterns for Object-Oriented Software Development*. Addison Wesley Longman.

Rayner, P., Wall, P. & Kruger, S. (2004). *Media Studies: The Essential Introduction*. Routledge, London, 2nd edn.

Rice, J. (1994). *Mathematical Statistics and Data Analysis*. Duxbury Press, 2nd edn.

Riehle, D. (1996). Describing and composing patterns using role diagrams. In *Proceedings of the 1996 Ubilab Conference*.

Riehle, D. (1997). Composite design patterns. In *OOPSLA '97: Proceedings of the 12th ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, vol. 32, 218–228, ACM Press.

Roberts, D. & Johnson, R. (1996). Evolving frameworks: A pattern language for developing object-oriented frameworks. In *Proceedings of the Third Conference on Pattern Languages and Programming*, vol. 3.

Rodgers, M., Sowden, A., Petticrew, M., Arai, L., Roberts, H., Britten, N. & Popay, J. (2009). Testing methodological guidance on the conduct of narrative synthesis in systematic reviews: Effectiveness of interventions to promote smoke alarm ownership and function. *Evaluation*, **15**, 49–73.

Ross, S. (1999). *Introduction to Probability and Statistics for Engineers and Scientists*. Academic Press, 2nd edn.

Schmidt, D. (1995). Using Design Patterns to develop reusable Object-Oriented Communication Software. *Communications of the ACM*, **38**, 65–74.

Schmidt, D., Stal, M. & Rohnert, H. (2000). *Pattern-Oriented Software Architecture Volume 2, Patterns for Concurrent and Networked Objects*. Wiley.

Schmidt, D.C. (1997). Applying design patterns and frameworks to develop object-oriented communication software. *Handbook of Programming Languages*, **1**.

Schmidt, D.C., Fayad, M. & Johnson, R.E. (1996). Software patterns. *Communications of the ACM*, **39**, 37–39.

Seaman, C.B. (1999). Qualitative methods in empirical studies of software engineering. **25**, 557–572.

Shelly, G., Cashman, T. & Vermaat, M. (2009). *Microsoft Office 2007: Introductory Concepts and Techniques, Premium Video Edition (Shelly Cashman)*. Course Technology.

Sommerville, I. (2007). *Software Engineering*. Addison-Wesley, 8th edn.

Struwig, M. & Stead, G.B. (2001). *Planning, Reporting & Designing Research*. Pearson Education South Africa, Forest Drive, Pinelands, Cape Town.

Summers, D., ed. (2006). *Longman Dictionary of Contemporary English (paperback) with CD-ROM*. Pearson ESL, Harlow, Essex, England, 4th edn.

Taylor-Powell, E. & Renner, M. (2003). Analyzing qualitative data. Tech. Rep. Bulletin G3658-12, University of Wisconsin-Extension, Madison, Wisconsin.

Thu, T. & Tran, H. (2007). A composite design pattern for object frameworks. In *COMPSAC '07: Proceedings of the 31st Annual International Computer Software and Applications Conference*, 521–526, IEEE Computer Society.

Torchiano, M. (2002). Documenting pattern use in java programs. In *Proceedings of International Conference on Software Maintenance (ICSM'02)*, 230–233.

Tukey, J.W. (1977). *Exploratory Data Analysis*. Addison Wesley, 1st edn.

Turner, M., Kitchenham, B., Budgen, D. & Brereton, P. (2008). Lessons learnt undertaking a large-scale systematic literature review. In *Proceedings of EASE 2008*, BCS-eWiC.

Unger, B. & Tichy, W. (2000). Do design patterns improve communication? an experiment with pair design. In *Proceedings International Workshop on Empirical Studies of Software Maintenance*, 1–5.

van Vliet, H. (2000). *Software Engineering: Principles and Practice*. Wiley, 2nd edn.

Vokáč, M., Tichy, W.F., Sjøberg, D.I.K., Arisolm, E. & Aldrin, M. (2004). A controlled experiment comparing the maintainability of programs designed with and without design patterns—a replication in a real programming environment. *Empirical Software Engineering*, **9**, 149–195.

von Mayrhauser, A. & Lang, S. (1999). A coding scheme to support systematic analysis of software comprehension. *IEEE Trans. Softw. Eng.*, **25**, 526–540.

Wendorff, P. (2001). Assessment of design patterns during software reengineering: Lessons learned from a large commercial project. In *Proceedings of 5th European Conference on Software Maintenance and Reengineering (CSMR'01)*, 77–84, IEEE Computer Society Press.

Wenzel, S. (2005). Automatic detection of incomplete instances of structural patterns in uml class diagrams. *Nordic J. of Computing*, **12**, 379–394.

Whitley, K.N. (1997). Visual programming languages and the empirical evidence for and against. *Journal of Visual Languages & Computing*, **8**, 109–142.

Wright, J. & Marsden, P. (2010). *Handbook of Survey Research*. Emerald Publishing Group Limited, 2nd edn.

Wydaeghe, B., Verschaeve, K., Michiels, B., Damme, B.V., Archens, E. & Jonckers, V. (1998). Building an omt-editor using design patterns: An experience report. In *Proceedings of Conference on Technology of Object-Oriented Languages, TOOLS*.

Yin, R. (2002). *Case Study Research: Design and Methods*, vol. 5 of *Applied Social Research Methods*. Sage Publications, Inc, 3rd edn.

Yuanhong, W., Hong, M. & Weizhong, S. (1997). Experience report: Using design patterns in the development of jb system. In *Proceedings of Technology of Object Oriented Languages and Systems: TOOLS 24*, 159–165, IEEE Computer Society Press.

Zhang, C. & Budgen, D. (2011a). A survey of experience about design patterns, paper in preparation.

Zhang, C. & Budgen, D. (2011b). What do we know about the effectiveness of software design patterns?, paper submitted to IEEE Transcations on Software Engineering.