Information and Software Technology 55 (2013) 822–835

Contents lists available at SciVerse ScienceDirect

Information and Software Technology

journal homepage: www.elsevier.com/locate/infsof

A survey of experienced user perceptions about software design patterns

Cheng Zhang [a], David Budgen [b,*]

[a] Anhui University, Hefei, China
[b] Durham University, Durham, UK

ARTICLE INFO

Article history:
Received 16 November 2011
Received in revised form 5 September 2012
Accepted 14 November 2012
Available online 29 November 2012

Keywords:
Software design patterns
Survey
Software design

ABSTRACT

Context: Although the concept of the software design pattern is well-established, there is relatively little empirical knowledge about the patterns that experienced users consider to be most valuable.
Aim: To identify which patterns from the set catalogued by the 'Gang of Four' are considered to be useful by experienced users, which ones are considered as not being useful, and why this is so.
Method: We undertook a web-based survey of experienced pattern users, seeking information about their experiences as software developers and maintainers. Our sampling frame consisted of the authors of all of the pattern papers that we had identified in a preceding systematic review of studies of patterns.
Results: We received 206 usable responses, corresponding to a response rate of 19% from the original sampling frame. Most respondents were involved with software development rather than maintenance.
Conclusion: While patterns can provide a means of sharing 'knowledge schemas' between designers, only three patterns were widely regarded as valuable. Around one quarter of the patterns gained very low approval or worse. These observations need to be considered when using patterns; teaching students about the pattern concept; and planning empirical studies about patterns.
© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Over the past two decades, the concept of the software design pattern has become an accepted part of the software engineering design lexicon. Design patterns seek to codify reusable experience about the way that a system, or part of it, can be organised, with the book by the 'Gang of Four' (abbreviated in the rest of this paper to *GoF*) being by far the most widely-known resource [1].

Since the seminal 2004 paper at ICSE by Kitchenham et al. [2], there has been a growing interest in creating an *evidence base* for software engineering. Evidence-based research has had a major impact upon practice, teaching and research in clinical medicine. This paradigm has also been applied successfully in other areas of healthcare as well as in such disciplines as social science and education, where the influence of human-based skills provide confounding factors similar to those encountered in software engineering studies [3]. The key tool for conducting evidence-based studies is the *Systematic Literature Review*, or SLR. Tertiary studies, in the form of systematic reviews of published SLRs addressing software engineering topics, have identified over 120 published reviews up to the end of 2009 [4–6].

The outcomes from SLRs of software engineering topics are often less conclusive than those occurring in clinical medicine, partly because of the human skill and experience factor, and also partly because there are often relatively few primary studies to draw upon (and few replications). However, there are a number of areas where the outcomes of SLRs have already demonstrated that the guidance derived from expert judgement, so often used to form the basis of software engineering practice, may not be fully supported, or even be contradicted, by the evidence [7–9].

We conducted a *mapping study* to examine the extent and form of the empirical knowledge that is available for software design patterns [10]. A mapping study is a form of SLR that has a much broader research question than is usual for an SLR, and is mainly concerned with identifying the extent and form of empirical knowledge available for a given topic [11]. The number of empirical studies that met the inclusion criteria was quite small—our final analysis included only 10 papers, describing 11 experimental studies, supplemented by seven observational 'experience' reports. All of the studies that we found addressed patterns that are catalogued in the *GoF* text. Although there are 23 patterns described by the *GoF*, the empirical studies only investigated a subset of these and only *Composite*, *Observer*, and *Visitor* had been studied very extensively.

Our mapping study found rather mixed evidence about the scope of 'usefulness' for the patterns that had been studied. 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. To investigate this question further, we therefore conducted a survey of

* Corresponding author.
E-mail address: david.budgen@durham.ac.uk (D. Budgen).

0950-5849/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.infsof.2012.11.003

experienced pattern users to identify those patterns from the set catalogued in the *GoF* text that are widely perceived to be of value, and hence likely to be used by developers and also which ones are little valued or used. Our research question for the 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?*"

Clearly, 'usefulness' can be interpreted in different ways depending upon the role of the user. In the survey, we sought to focus upon the use of patterns for software development and maintenance, and worded our questions around those roles. In the rest of this paper we briefly discuss the subject matter (design patterns); describe the design and conduct of the survey; present our results and analysis; and discuss the limitations of our survey as well as its basic implications.

2. Background

Two issues that influenced the design of our survey were the characteristics of software design patterns and the findings from our mapping study.

2.1. Design patterns

The concept of a design pattern stems from the work of the architect Alexander et al. [12], with his definition of a pattern being expressed as follows:

> "*Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.*"

In an early study of software design activities conducted by Adelson and Soloway [13], they used the term 'labels for plans' to describe a similar concept—denoting a process by which a designer would recognise and 'label' a sub-problem that they knew how to solve, leaving them free to focus upon the task of addressing the less familiar aspects of a design task. In her study of cognitive aspects of software design, Détienne uses the term 'knowledge schema' to describe a broader concept—where a schema is described as a 'knowledge structure' that represents "generic concepts stored in memory" [14].

Software design patterns can therefore be viewed as a realisation of Alexander's pattern concept, using a form that codifies and externalises expert knowledge schema that have been derived from software design practice [15]. Implicitly therefore, design patterns are likely to be more valuable to experienced designers (who can use them to extend their own knowledge schema) than to novice designers, a view that was supported by the qualitative observations identified in [10].

Although many software design patterns have been catalogued, the concept (and specification) of a design pattern is widely associated with the definitive text by the *GoF*. The empirical studies that we identified while conducting our mapping study were almost exclusively concerned with the patterns from this book and hence our survey was confined to this set of 23 patterns.

2.2. The mapping study of design patterns

In clinical medicine, SLRs usually aim to aggregate the outcomes from *Randomised Controlled Trials* (RCTs), where the use of double-blinding and the role of the participants as *recipients* of the experimental treatment makes it feasible to use statistical techniques to synthesise the outcomes of a set of studies. (*Double-blinding* indicates that neither the recipients of a treatment, nor those administering it, are aware of who is in the treatment group and who is in the non-treatment control group.) For software engineering, where the treatment being studied usually involves the participants in performing skill-related activities, double-blinding is usually impractical, which can make the synthesis of results from the studies much more challenging [16]. Most SLRs seek to aggregate outcomes from more rigorous forms of primary study, such as experiments and quasi-experiments [17], but for studies of design activities in particular, it is not unusual to also include case studies [18,19] and to perform a narrative synthesis.

Our mapping study investigating empirical studies of software design patterns is reported in detail in [10]. We searched for papers using electronic databases and also conducted a manual search of major journals (to ensure that we were not missing papers that used different keywords). The search process located 611 candidate papers, but after a rigorous inclusion/exclusion process we were left with only 10 papers describing 11 experimental studies, and no case studies. To augment the available data we found it necessary to include what we have termed 'experience' papers that are based upon relatively systematic observation in the field. We included only those that provided a clear link between their observations and their conclusions [20], which provided a further seven studies.

However, even within this group of observational studies, with the exception of a particularly well-reported paper by Wendorff [21], the links were often poorly articulated. We subsequently drew upon our experience (and problems) with extracting data from this latter group of papers when designing our survey, in order to seek more information about the links between a respondent's views and the experiences that underlay those.

Only three patterns were investigated in more than a few studies: *Composite*, *Observer* and *Visitor* each being used in seven studies. Eight patterns were not investigated in any studies at all. Even for the three that were studied more extensively, there was considerable variation in the outcomes of different studies. Only one paper gave a rationale for the choice of pattern being investigated, and the overall impression was that the choice of patterns for the experimental papers was largely determined by the particular software artifacts selected for the studies.

3. Research method

A survey provides a well-established means for eliciting knowledge about a topic from the expertise and experience that is contained largely within a reasonably well-defined community. Fink suggests that surveys can be categorised as either *descriptive* or *experimental* designs, and provides further sub-categories of these [22]. In this particular context, we considered our survey to have elements of both forms. The descriptive element was provided through the profiling of usefulness of the different patterns, as provided by the expert respondents (with this aspect of the study having a *case control* form [23]). The experimental aspect was provided by comparing the ways in which different groups viewed the patterns (categorised as *concurrent study in which participants are not randomly assigned to groups*). For this purpose, the respondents could be grouped along two different axes: one being their years of experience with using design patterns; the other being their current role. In both cases, individuals could not be allocated randomly to a group.

Kitchenham and Pfleeger have discussed how surveys can be used to address software engineering topics, and some of the issues that need to be considered when using a survey [24,23,25–28]. Conducting a survey presents a range of challenges. Design issues that arise in planning a survey include: identifying (and accessing)

the relevant population; using appropriate forms of sampling where the population is large; obtaining an adequate response rate; and avoiding bias in the questions. In addition, a survey is essentially a 'one-off' process, with limited scope for repeating a study should its design prove faulty, or for replication. To design our own survey, we began by writing a research protocol using a template based upon the articles by Kitchenham and Pfleeger.[1] In the rest of this section we discuss the main decisions involved in developing the protocol. In the following section we then go on to describe how the survey was conducted, and in what ways it diverged from the plan.

3.1. Research question

As identified in the introduction, our overarching research question was: "*which design patterns from the GoF do expert pattern users consider as useful or not useful for software development and maintenance, and why?*". In particular, we wished to identify which patterns were actually used to any significant extent.

3.2. Design

The structure employed for data collection was motivated by the research question. After an initial set of questions used to obtain a profile of a respondent and their expertise, the first technical question was a 'rating' one, used to obtain information about the extent of the respondent's familiarity with, and knowledge about, all of the 23 patterns. For each pattern, the respondent was asked to select a 'rating' from a four-point Likert scale (*Very Useful*, *Useful*, *Not Very Useful*, *Not at All Useful*) or to select a *Little or No Experience* option if they felt unable to use one of these. Because the aim of our survey was to assess the usefulness of these patterns, we used the 'forced choice' form of Likert scale, with no 'neutral' point, in order to encourage respondents to make a positive or negative statement. The question was presented as a matrix of patterns and ratings, and while respondents could rate patterns in any order, they could not proceed to the next question until a value had been selected for every pattern.

The second technical element was two 'ranking' questions that asked the respondent to identify those patterns that they considered most useful, together with their reasons for choosing each one, and then those patterns that they considered least useful (and why). The option for entering associated free text information about the reasons for choosing a pattern provided scope for a respondent to supply the (explanatory) causal link between their choices and their experiences. Guidance on the design of *ranking* questions of this form is that ranking long lists can produce highly unreliable results and that it is more effective to ask respondents to identify the two or three items at the top of their list [29]. We therefore asked respondents to identify up to three patterns for both categories.

A key element in designing a survey is to determine the population that is to be studied (the *target population*). For our study, we ideally wanted to survey a representative set of software developers and maintainers who possessed experience of using patterns and maintaining systems that had been developed with their use. However, obtaining access to this group presented a major problem—because we could not identify an obvious forum through which they could be identified or contacted. Indeed, since software development is a world-wide activity, and this particular community is defined in terms of their specific skills and knowledge, rather than by location or through membership of any formal organisation, there is no way of knowing the size of the target population either.

Our solution to this problem was to use a 'surrogate' group to which we did have access, namely the set of authors of papers about patterns. The systematic search process of our mapping study had generated a comprehensive list of papers about design patterns, and so extracting the details of the authors from the complete set of papers provided a set of 882 names after removing duplicates, together with e-mail contact details.

This list therefore provided both our *sampling frame* and also the (cluster-based) sample, as we decided to mail all of the authors on the list in order to obtain the maximum confidence level in the results. A typical level of response to surveys is of the order of 10%, so after making allowance for expired contact details it was anticipated that we might obtain of the order of 50–100 responses. In addition, because the contact information was for electronic access (e-mail) we planned to send a link to the data collection form to each recipient by using e-mail.

One method for improving response rates is to send one or more follow-up messages to non-respondents after a suitable interval. In designing our survey, we decided that we would send only one follow-up message, and that this would be sent approximately 2 weeks after the original request.

4. Conduct of the survey

Here we describe the implementation of the survey.

4.1. The survey form

Two key elements of this were the development of the actual survey instrument and the means of administering it. Because the second influenced the organisation of the first, we begin by describing that particular choice.

The survey was administered through the services of a well-established commercial site (*SurveyMonkey*). Benefits of using this site included the provision of secure access, greater user confidence (many on-line surveys had been organised through this site), and the availability of tools for managing the survey process and analysing the outcomes.

The choice of *SurveyMonkey* then determined the range of graphical options that could be employed for implementing the closed questions, such as radio buttons and drop-down menus. Examples from the final set of questions used are provided in Table 1, involving a mix of technical and demographic information. Closed questions were used for much of the survey, making it easier and faster to complete, but an open format was used wherever we wished to seek explanatory information. Where we were seeking information about a subjective issues such as a respondent's assessment of usefulness, we used a four-point Likert scale together with an option that could be used to indicate that the respondent was not in a position to provide an opinion.

Once developed, the form was reviewed by two external assessors (a 'dry run'), following which a number of changes were made in order to improve presentation and clarity. In particular, this included adding a continually updated 'progress report' to keep the respondent informed about how far they had progressed through the form and hence how much was left to complete.

4.2. Administering the survey

To organise the survey, and to help manage the issue of reminders, requests were sent out to authors from the list in batches (usually 50). For each batch, a short reminder message was then sent

[1] Templates for different forms of empirical study are available at: http://www.ebse.org.uk.

Table 1
Selected survey questions.

1	*Request for name and e-mail contact*
2	Which of the following best describes your primary role during software development? (commercial software developer; software researcher; software teacher; student of computing)
3	Highest degree you have earned? (Associate degree; Bachelor's degree; Masters; PhD or equivalent; Other)
4	How many years of experience do you have with Object-Oriented development? (Less than 3 years; 3–5 years; 6–10 years; 11–15 years; Over 15 years)
5	How many years of experience do you have with working with design patterns? (Less than 3 years; 3–5 years; 6–10 years; 11–15 years)
6	Have you written any patterns (or rewritten any existing patterns)? (Yes; 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)
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 (SECOND; THIRD) 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 onto the next set of questions.)
9	What type(s) of software have you developed or maintained with this pattern (You can check more than one.) (Productivity/business software; Graphic design and Multimedia; Home/Personal/Education; Distributed Systems (such as web-based application); System Software (e.g. Operating system, Middleware))
10	What was the size of the system? KLOC (up to 100; 100–250; 250–500; above 500)
11	What was the level of abstraction involved in using this design pattern? (Design; Code; Both)
12	In what stage(s) in the life-cycle of the system(s) did you work with this design pattern? (Development; 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, and then we use a similar structure to identify patterns that were not found to be useful. 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.

2 weeks later to those who had not responded (dates were changed according to the group).

With the passage of time since the publication of many of the papers about patterns, some of the e-mail addresses extracted from the papers were no longer current. In total some 877 requests were mailed out, and on the basis of returned e-mails, 196 were identified as having been sent 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 (snowball sampling). Since we knew the details of the original list, we were able to distinguish between those asked directly and those who had the request copied to them. (This information was purely used for this purpose and apart from this, all responses were treated anonymously.)

The invitation was also distributed to three research-oriented mail-lists that were identified while conducting the survey. The sizes of these are not known, making it impossible to calculate response rates. However, their use did generate some additional responses from a group likely to have similar backgrounds to those on the original list.

4.3. Divergence from the protocol

The one feature of the conduct of the survey that diverged from the original protocol was the inclusion of the members of two additional groups (one of which we were unaware of when planning the original survey distribution) in our sampling. As indicated above, we could distinguish these responses from those sent directly to authors, and by using different time intervals for surveying the different groups, we were also able to distinguish those where the authors had passed our survey details on to colleagues from those received from members of the mail-lists.

As a consequence, our survey 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 mail-lists formed a self-selection sample.

5. Profile of the respondents

We first present the 'raw' numbers related to the three groups of respondents. We refer to these as *Authors*, *Snowball* (of people known to authors) and *Mail-list* (those contacted via the three research mail-lists). We then present a statistical analysis of the profiles of these to see how far they differ.

5.1. The responses

In total, 227 responses were received, which was well in excess of expectation. A number of responses (21) were unusable because the respondents only entered the 'demographic' and 'administrative' information (questions 1–6, 45–47) and so these were removed from the dataset, leaving a total of 206 responses. Table 2 shows how the responses were distributed among the three groups of respondents.

If the invalid addresses are removed from the set of responses (on the basis that as these people did not receive the request they could not have responded), then the response rate obtained for the group originally targeted (authors) is 128/681 or 19%. While it is not possible to know how many requests to others were made by the authors, if we simply include the responses obtained through snowball sampling and again discard the invalid addresses, then the resulting response rate for the extended group (authors and people they know) is $169/(681+53)$ or 23%. (Although this response rate is good, because the sampling frame was relatively small, it still equates to a confidence level a little below the desired 95% ±5%.) Given that there is no way of knowing the size of sampling frame for the research mail-lists, it is not possible to calculate a meaningful response rate for this third group.

Where relevant, we have separated the counts for the different groups, as well as providing counts for the complete set of 206 respondents.

The profiles of the respondents in terms of their education and their current primary roles are summarised in Tables 3–5.

Given that the original sampling frame used was based upon authorship of papers, it is perhaps not surprising that over half of the respondents were researchers, teachers and students. Nonetheless, the proportion who considered themselves to be developers was still significant. The overwhelming majority were university graduates of some form, with a large proportion having PhDs, particularly among researchers and teachers. This educational profile might again have been influenced by the nature of the sampling frame, but in the absence of any known profile for patterns users or software designers, there is no way of determining how representative it is.

Table 6 provides a profile of the experience of the different groups of respondents with object-oriented development in general. More than half had over 10 years of experience with OO development. Table 7 provides a similar profile for experience with OO patterns. Given that design patterns did not emerge until the early 1990s, it is therefore not surprising that experience with patterns is less extensive, although still substantial.

Since authorship of papers about patterns formed the basis for the main sampling frame, we did expect to find a substantial degree of experience with writing patterns. This was supported by the responses to Question 6 (''Have you written or rewritten any patterns?''). As shown in Table 8, over half of the respondents had pattern authoring experience, and only for the *Snowball* group was the proportion more balanced.

Finally, for each pattern that a respondent identified as being either useful or not useful, we asked how they had obtained experience with that pattern, whether this was through either development or maintenance activities. Tables 9 and 10 show that, regardless of group, this had been predominantly through the development of systems (Dv) rather than through maintenance (Mt)—a point that we will return to later.

5.2. Statistical analysis of the profiles

We performed a descriptive statistical analysis to answer the question ''*how far did the three groups actually differ?*'' and hence how far we needed to treat their responses separately for the purpose of analysis. Our statistical analysis used both parametric and non-parametric tests.

5.2.1. Experience of OO development

While the profiles for the three groups do differ, it is noticeable that the *Snowball* group has a profile very like that of the *Mail-list* group. For testing purposes our null hypothesis was: *Hypothesis 1: the respondents from the three groups will not be different in terms of their experience of software development*. Applying the Kruskal–Wallis test to the responses to Question 4 showed that the result had a significant difference ($p < 0.05$) and hence the null hypothesis was rejected. We then applied Tamhane's T2 test [30], to look at the variance between the groups, with the dependent variable being ''how many years of experience do you have with Object-Oriented development'', and with the outcomes from this being shown in Table 11. For this table, the second column indicates the difference between the mean level of experience for each group in the pair, while the third compares the mean experience of the combined group with that of the whole population being sampled. For the confidence interval, a '(∗)' indicates that the interval does not contain zero. For the third row, the p value of 0.996 between

Table 2
Profile of responses.

	Authors	Snowball	Mail-list
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 3
Profile of respondents: education.

Highest degree	Authors	Snowball	Mail-list	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 4
Profile of respondents: primary roles.

Category	Authors	Snowball	Mail-list	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 5
Profile of respondents: education versus role.

Degree	Primary role			
	Developer	Researcher	Teacher	Student
Associate	1	0	0	0
Batchelor's	30	1	0	2
Masters	31	8	2	4
PhD	16	70	37	1
Other	3	0	0	0
Total	81	79	39	7

Table 6
Profile of respondents: experience with OO development.

Length of experience	Authors	Snowball	Mail-list	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
Profile of respondents: experience with OO patterns.

Length of experience	Authors	Snowball	Mail-list	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 8
Pattern authoring experience.

Pattern author?	Authors	Snowball	Mail-list	Total	
				(#)	(%)
Yes	79	20	23	122	59.2
No	49	21	14	84	40.8

Table 9
Source of experience for 'useful' patterns.

Group	Authors		Snowball		Mail-list		Total	
Phase	Dv	Mt	Dv	Mt	Dv	Mt	Dv	Mt
Choice 1	99	5	25	4	26	3	150	12
Choice 2	87	6	21	4	22	3	130	13
Choice 3	80	8	17	3	17	4	114	15
Total	266	19	63	11	65	10	394	40

Table 10
Source of experience for 'not useful' patterns.

Group	Authors		Snowball		Mail-list		Total	
Phase	Dv	Mt	Dv	Mt	Dv	Mt	Dv	Mt
Choice 1	23	1	2	3	5	2	30	6
Choice 2	17	0	2	1	4	1	23	2
Choice 3	13	0	1	0	4	0	18	0
Total	53	1	5	4	13	3	71	8

the Snowball group and Mail-list group is greater than 0.05 and hence there is no significant difference between these two groups. The standard deviation also indicates generally good agreement with the whole population, but given the small size of this, all three pairing show good agreement, as might be expected.

5.2.2. Experience with using design patterns

We undertook the same process for the profiles of the three groups concerning patterns experience. Here our null hypothesis was: *Hypothesis 2: the respondents from the three groups will not be different in terms of their experience of using design patterns.* Again, applying the Kruskal–Wallis test to the responses to Question 5 showed the result had a significant difference ($p < 0.05$) and so the null hypothesis was rejected. Applying the Tamhane T2 test again, with the dependent variable being "how many years of experience do you have of working with design patterns", produced the results shown in Table 12. Again, these show a difference between the *Authors* and the other two groups, but no significant difference between the *Snowball* and *Mail-list* groups ($p = 0.414$).

5.2.3. Effect of experience with using patterns and assessment of usefulness

Having established that we effectively had two groups of respondents in terms of the degree of experience with OO design and patterns use, we then investigated whether any differences between the responses of the groups to later questions simply reflected the fact that the *Authors* group contained more experienced respondents, rather arising from the members having any specific difference of perceptions about patterns.

We compared the profiles for the choices made in response to Question 7 (perceived usefulness of each pattern from the *GoF*) against the profiles of experience provided from Question 5. To do so, we combined the counts for the positive votes (very useful (VU) and useful (U)) and for the negative votes (not very useful (NVU) and not useful (NU)) as the boundaries between these were open to different interpretations, so in effect reducing the responses to a two-point scale. Tables 13–16 show the counts for the four bands of experience with using patterns, together with the contribution of each term to the overall χ^2 test for independence.

Here, our null hypothesis was: *Hypothesis 3: There will not be a difference between the assessments from the Author's group and the merged Snowball and Mail-list groups for patterns assessment.* We conducted a χ^2 test with two degrees of freedom and an α value of 0.05. Only Table 14 shows a value below that required (5.991), so that while the null hypothesis is accepted for this group, it is rejected for the others.

None of the other three groups reached the 0.05 level of significance in spite of the total number of responses for each of the groups being very large (690, 2001 and 1196 for the less than 3 years group, 6–10 years group, and 11–15 years group respectively). However, on closer examination, the larger contributions stem mainly from differences in the small number of negative votes or "little experience" votes. (In all cases, the observed number of positive votes was very close to the expected number.) In the less than 3 years group there was only 1 negative vote among the *Authors* group, rather than the expected 7.9, which contributed 6.0 to the χ^2 value. For the 6–10 years group there were relatively few negative votes among the merged group, i.e. 56 instead of the expected 73, which contributed 4.1 to the χ^2 value. In the case of the most experienced participants, the merged group had a relatively low number of "little experience" responses, 11 rather than the expected 20.2, contributing 4.2 to the χ^2 value.

Table 11
Multiple comparison for Hypothesis 1 (experience of software developers).

Group (i)	Group (j)	Mean Difference ($i - j$)	Std. error	Significance	95% Confidence interval	
					Lower bound	Upper bound
Authors	Snowball	1.134(*)	0.226	0.000	0.58	1.69
Authors	Mail-list	1.190(*)	0.211	0.000	0.67	1.71
Snowball	Mail-list	0.057	0.287	0.996	−0.65	0.76

Table 12
Multiple Comparison for Hypothesis 2 (experience with using design patterns).

Group (i)	Group (j)	Mean Difference ($i - j$)	Std. error	Significance	95% Confidence interval	
					Lower bound	Upper bound
Authors	Snowball	0.879(*)	0.180	0.000	0.43	1.32
Authors	Mail-list	1.199(*)	0.166	0.000	0.79	1.61
Snowball	Mail-list	0.320	0.227	0.414	−0.23	0.87

Table 13
Comparison between groups for respondents having less than 3 years experience with patterns.

Assessment	Authors		Merged	
	Count	χ^2 cont.	Count	χ^2 cont.
Pos (VU + U)	38	1.580	272	0.176
Neg (NVU + NU)	1	6.027	78	0.670
Little experience	30	0.000	271	0.000
χ^2 value				8.453

Table 14
Comparison between groups for respondents having 3–5 years experience with patterns.

Assessment	Authors		Merged	
	Count	χ^2 cont.	Count	χ^2 cont.
Pos (VU + U)	234	0.933	274	0.711
Neg (NVU + NU)	39	0.793	65	0.604
Little experience	95	0.675	144	0.514
χ^2 value				4.230

Table 15
Comparison between groups for respondents having 6–10 years experience with patterns.

Assessment	Authors		Merged	
	Count	χ^2 cont.	Count	χ^2 cont.
Pos (VU + U)	1011	0.137	358	0.429
Neg (NVU + NU)	234	1.386	56	4.096
Little experience	250	0.119	92	0.352
χ^2 value				6.519

Table 16
Comparison between groups for respondents having 11–15 years experience with patterns.

Assessment	Authors		Merged	
	Count	χ^2 cont.	Count	χ^2 cont.
Pos (VU + U)	781	0.000	144	0.002
Neg (NVU + NU)	111	0.470	29	2.584
Little experience	120	0.755	11	4.155
χ^2 value				7.966

On the basis of this, we therefore concluded that the profiles of choices against experience for these two groups could be considered as being sufficiently similar for them to be considered as coming from the same overall population. Hence for the rest of our analysis we treated all 206 respondents as being from one group.

6. Results

The technical element of the survey was in two parts. Question 7 addressed the *descriptive* element of the survey by asking respondents to provide a profile of rating values for all of the 23 patterns. The remaining questions addressed the *experimental* element of the survey and were in the form of ranking actions, concerned with profiling knowledge and experience for up to three 'most favoured' and three 'least favoured' patterns.

6.1. Overall profile of usefulness (rating)

Since we only included responses that answered this question, we had values from all 206 respondents. The question was presented in the form of a 'grid', with the 23 patterns listed down the left side of the screen and the rating scales displayed across the rows. Apart from convenience in presentation, since all patterns were viewed at once, this choice of layout was intended to reduce any influence upon the following ranking questions. The scale value was expressed in terms of a simple 4-point Likert scale (very useful (VU); useful (U); not very useful (NVU); not at all useful (NU); together with an option for recording little or no experience of using this pattern (NE)). Table 17 provides a summary of the ratings, giving two counts for each pattern and each scale value. The first count represents the responses from all 206 respondents, while the second (in italics) is the count for the subset of 176 respondents who indicated that they had more than 3 years of experience with using patterns. In the question (and also in the table), patterns were listed in the same order as in the *GoF* text, and hence were listed in the order: creational (C); structural (S) and behavioural (B). The one exception was *Chain of Responsibility*, which appeared at the end of the behavioural set, rather than at the beginning.

Table 17
Summary of perceived usefulness (rating).

Pattern	VU	U	NVU	NU	NE
Creational patterns					
Abstract Factory	83	84	13	3	23
	74	*74*	*10*	*2*	*16*
Builder	28	78	38	3	59
	26	*72*	*31*	*2*	*45*
Factory Method	100	73	12	2	19
	87	*63*	*10*	*2*	*14*
Prototype	24	65	42	4	71
	23	*58*	*39*	*4*	*52*
Singleton	102	52	30	11	11
	89	*43*	*27*	*10*	*7*
Structural patterns					
Adapter	99	76	6	3	22
	89	*66*	*4*	*3*	*14*
Bridge	36	88	18	4	60
	34	*81*	*12*	*4*	*45*
Composite	115	51	11	3	26
	110	*41*	*9*	*3*	*13*
Decorator	72	73	20	4	37
	65	*63*	*19*	*4*	*25*
Facade	88	70	16	2	30
	78	*63*	*15*	*2*	*18*
Flyweight	15	63	37	9	82
	14	*57*	*34*	*9*	*62*
Proxy	90	70	9	1	36
	85	*61*	*8*	*1*	*21*
Behavioural patterns					
Command	72	75	17	2	40
	64	*71*	*11*	*2*	*28*
Interpreter	22	56	36	13	79
	22	*51*	*31*	*12*	*60*
Iterator	104	62	11	2	27
	93	*55*	*11*	*1*	*16*
Mediator	35	66	36	2	67
	34	*59*	*31*	*2*	*50*
Memento	15	52	36	14	89
	14	*48*	*32*	*12*	*70*
Observer	127	53	9	2	15
	115	*43*	*7*	*2*	*9*
State	57	65	30	4	50
	54	*60*	*27*	*3*	*32*
Strategy	83	69	15	3	36
	75	*60*	*15*	*2*	*24*
Template Method	79	60	18	4	45
	75	*52*	*15*	*4*	*30*
Visitor	77	66	20	8	35
	69	*59*	*17*	*7*	*24*
Chain of Responsibility	35	87	25	5	54
	35	*78*	*21*	*5*	*37*

The spread of values between the responses from the complete set of respondents and the 'more experienced' subset (excluding those with less than 3 years of experience with using patterns) shows no particular characteristics beyond, as might be expected, a slight concentration of 'votes' from the less experienced under the heading of 'little or no experience'.

We should probably not infer too much from the small numbers in the 'not at all useful' column, because making that choice represents quite a strong statement. However, if we aggregate the numbers in the first two columns ('votes' representing a positive view about a pattern) then we find a number of patterns that fail to obtain the 'approval' of more than 50% of the respondents (both lines). These are *Prototype*, *Flyweight*, *Interpreter* and *Memento*.

Equally, there are a small number of patterns that are clearly very well known and liked, with *Observer* and *Composite* in particular being both highly valued as well as widely-known. While *Singleton* is marginally the most widely-known, opinions about its value are rather more mixed, particularly among the respondents with greater experience.

There are also a small number of patterns that are unfamiliar to at least 25% of either sample: *Builder*, *Prototype*, *Bridge*, *Flyweight*, *Interpreter*, *Mediator* and *Memento*. All of these also have correspondingly low counts for 'very useful', suggesting that even those familiar with them find them of less value than the others (the overlap with the set getting more negative votes is perhaps not surprising). Together, these form a little over 30% of the patterns from the *GoF* set.

We also looked at how the 'votes' were used by the respondents according to their current roles (researchers, teachers and developers). We counted the number of 'very useful' and 'useful' votes from each group and aggregated these. The outcome is shown in Table 18. Informally, the values shown there indicate that both voting 'levels' were used more or less equally by all three groups and that while, from the averages, teachers identified slightly more patterns as being useful to some degree, the difference did not appear to be very marked. As it is quite possible for any one person to perform all of these roles at different times, this is perhaps not completely surprising.

6.2. Patterns considered useful (ranking)

One-hundred and sixty-four respondents provided some information in response to these questions. Those not responding were roughly proportional to the group sizes (25 authors, 10 from the *Snowball* group and 7 from the *Mail-list* 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. We did not attempt to analyse the ordering of these preferences in any way, simply considering each one as being a 'vote' for that particular pattern. Table 19 lists the six 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 characterised by mainly picking up 'third votes'. There also appears to be no distinction between the types of pattern preferred, all three forms (creational, structural and behavioural) are equally represented.

At the other end were the patterns that generated least enthusiasm from respondents. Table 20 shows the group of patterns with the lowest counts. With the exception of *Chain of Responsibility*, the elements on this list are consistent with those in the lowest scoring group in Table 17, indicating consistency of responses between the rating and ranking questions. The slightly anomalous position of *Chain of Responsibility* can be explained by its rather specialist nature—leading it to be perceived as generally useful, but not sufficiently so for it to be ranked very highly. One respondent did observe that "*Chain of Responsibility is a useful pattern, but places where it can be used are few and far between, so I see it as not as useful*". The spread across the different forms is also quite closely proportional to the ratios of the different forms in the *GoF* set).

6.3. Patterns not considered useful (ranking)

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 listed three patterns, 9 listed two and 20 listed only one. Of these 58 respondents, only four had less than 3 years experience. Based upon these responses, Table 21 lists the four patterns that were considered the ones to be most avoided.

While it is not entirely surprising to find some overlap with Table 20, and indeed, two of the patterns, *Flyweight* and *Memento* do appear in both, it is perhaps surprising to find *Visitor* and *Singleton* listed here, especially as they both also feature in Table 19. However, this could well reflect the extent to which the successful use of some patterns is particularly dependent upon context. Indeed, if we examine the experiences from the mapping study, then we find that the findings from the experimental and observation studies also provide some contradictory views about these pat-

Table 18
Deployment of usefulness votes by role.

Sample	Researchers	Teachers	Commercial developers
Authors			
VU	541	335	166
U	522	306	194
Snowball			
VU	47	10	178
U	35	9	184
Mail-list			
VU	20	0	231
U	25	0	241
Total	1190	660	1194
Average	15	17	15

Table 19
Most highly favoured patterns.

Pattern	Choice 1	Choice 2	Choice 3	Total
Observer (B)	28	17	14	59
Composite (S)	25	11	12	48
A. Factory (C)	17	10	7	34
Singleton (C)	7	7	17	31
Visitor (B)	13	14	4	31
Facade (S)	9	14	7	30

Table 20
Least favoured patterns.

Pattern	Choice 1	Choice 2	Choice 3	Total
Memento (B)	0	0	0	0
Prototype (C)	1	0	0	1
Interpreter (B)	1	0	1	2
Flyweight (S)	2	0	0	2
Chain of Responsibility (B)	2	0	2	4
Builder (C)	2	1	2	5
Mediator (B)	3	1	1	5

Table 21
Patterns not considered useful.

Pattern	Choice 1	Choice 2	Choice 3	Total
Flyweight (S)	14	7	2	23
Singleton (C)	11	2	1	14
Visitor (B)	4	5	3	12
Memento (B)	2	6	2	10

terns. In the case of *Singleton*, we might also note that in the answers to Question 7, it was evident that the pattern was perhaps more well-known than valued.

As with the previous set of questions, the experiences were largely derived from development activities rather than maintenance.

6.4. The combined responses

Fig. 1 shows the proportion of all positive and negative votes for each pattern—those above the line are the percentage of all votes for the pattern as being useful, those below are the percentage of all votes for the same pattern as not being useful. The percentages are respectively in terms of all positive votes (389) and all negative votes (113). Above and below the bars for each pattern are the actual number of votes in parentheses.

When the results are presented in this way, the three patterns that clearly have the most varied 'spread' in terms of a substantial and relatively even mix of positive and negative views are *Singleton*, *Visitor* and *Flyweight*, all of which have been noted in the tables above. As such, the visualisation does highlight the degree of ambivalence about *Singleton* and *Visitor* (and to a lesser extent, about *Facade* too).

6.5. Responses categorised by role and experience

For the experimental element of the survey, we also looked at how the votes were allocated against the current role of the respondent. For this purpose, we used two groups: the developers; and then a combined 'academics' group of researchers and teachers (largely because these two roles are often not clearly distinguished). Also, we included only the votes from respondents with more than 3 years experience of using patterns. The resulting values are shown in Figs. 2 and 3. To aid comparison, we have organised both of these alphabetically and normalised them by presenting them as percentages.

The profiles are broadly similar, indicating that the perceptions of the two groups are in general agreement, which reinforces our earlier decision to treat the sample as being from one overall population. While they look different for a number of patterns, using a χ^2 test to compare the profiles for a number of patterns, found little difference. At a 0.05 level of significance, there was a small difference concerning *Bridge*, favoured more by Academics, and at just below that level of significance there was a small difference for *Command*, favoured more by Developers. However, as these garnered a total of 14 and 17 votes respectively, the distinctions should not be over-stressed.

As a consistency check, we tabulated the results from the rating question (Q7) for the 61 developers who had more than 3 years experience with using patterns, and these are shown in Table 22 (we have again aggregated the two positive and negative categories).

These are clearly consistent with the rankings for *Facade*, *Proxy* and *Command*, and obviously, these patterns are ones that are valued by developers. *Bridge* is less well known, and as usual, the views on *Singleton* are spread widely. While direct comparison between the outcomes from the two forms of question is impractical, Table 22 suggests that overall, the two show largely consistent views about these patterns.

7. The qualitative data

For each pattern ranked by a respondent as being in their 'three most/least useful' group, there was an associated open-ended question that asked them to share their experiences. (We had realised the need to link experiences with opinions from our use of 'experience' papers in the mapping study—where very few authors provided this information.) Most participants who did respond to these questions described their experiences and the characteristics of the design patterns in simple words and brief phrases.

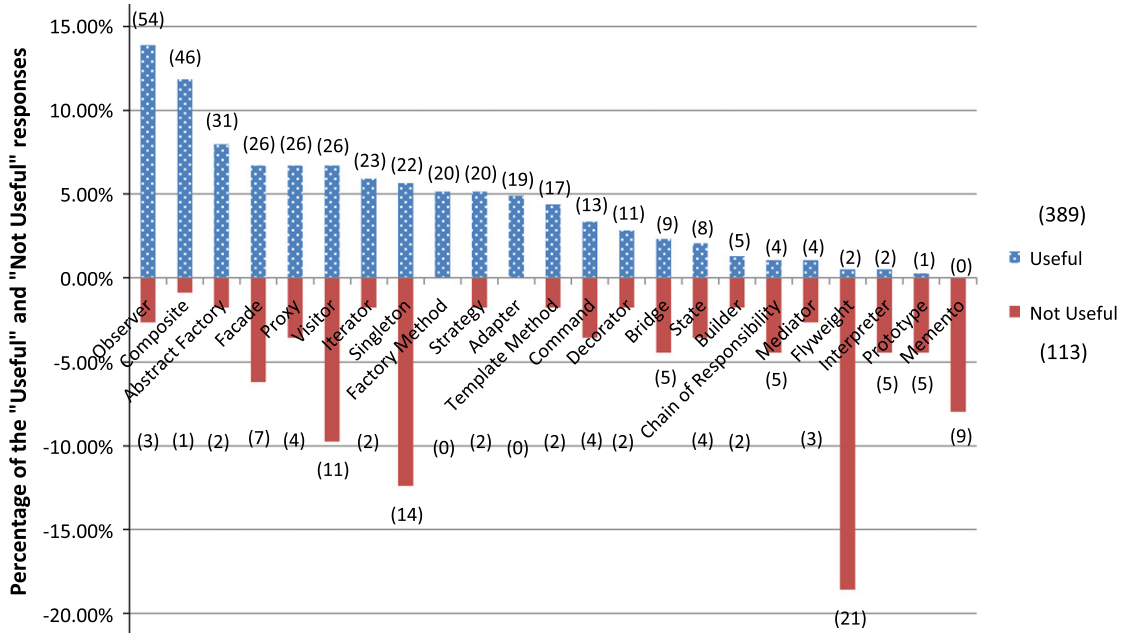

Fig. 1. Proportions of positive and negative votes for each pattern.

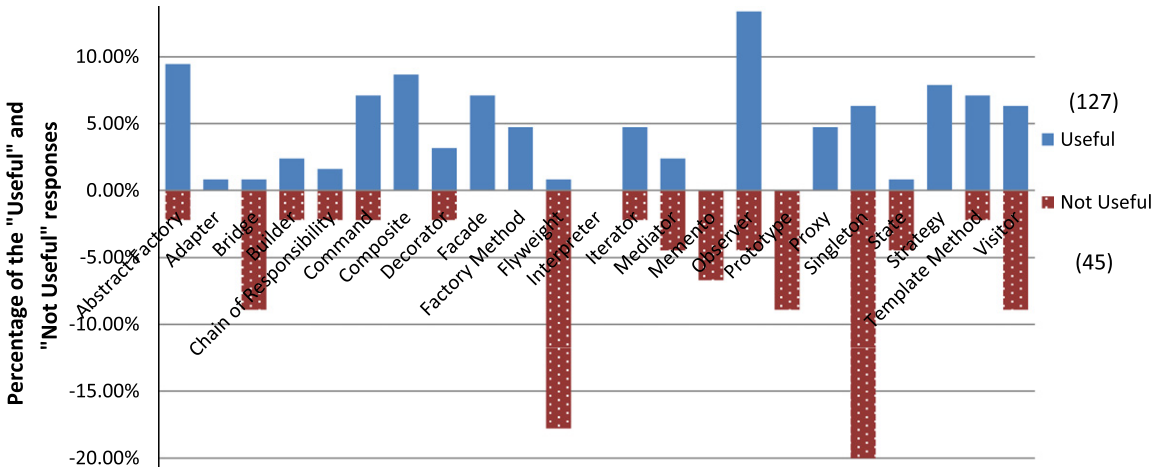

Fig. 2. Developer allocations of positive and negative votes by pattern.

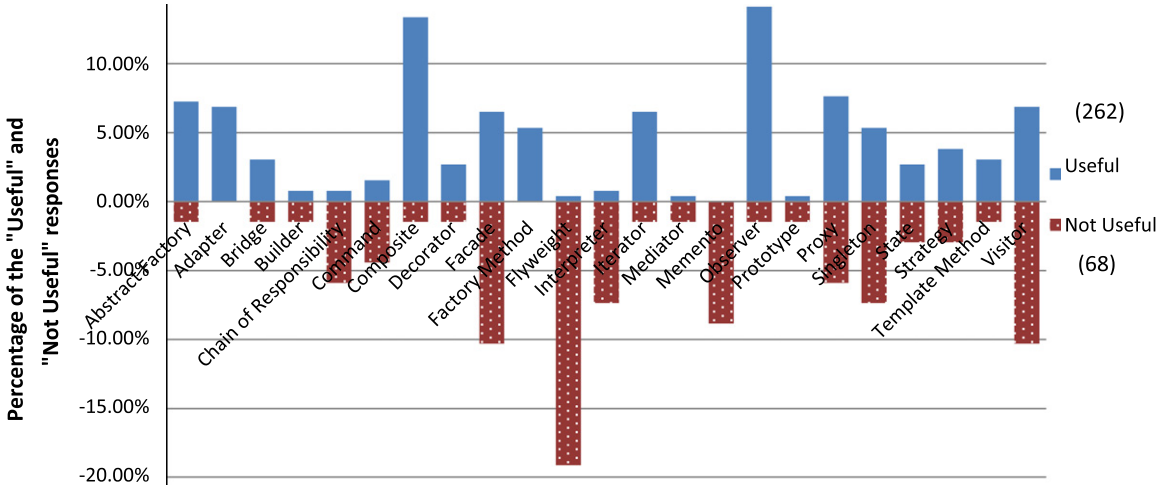

Fig. 3. Researcher/teacher allocations of positive and negative votes by pattern.

Table 22
Votes from Q7 for more experienced developers.

Pattern	Positive	Negative	Little experience
Bridge	36	7	18
Command	45	5	11
Facade	53	3	5
Proxy	54	2	5
Singleton	44	16	1

Unfortunately, we received relatively few comments. There were 266 comments associated with the 445 'positive' votes and 72 comments associated with the 125 'negative' votes. For the second group in particular, slightly more than half provided little explanation for their views, and few cited specific design examples or experiences that had caused them to rank the pattern negatively, although several did identify what they considered to be simpler alternatives to using that particular pattern.

In the rest of this section, we report comments about the *Visitor* and *Singleton* patterns. There are two reasons for this choice. one is that the quantitative results indicated a strong element of ambivalence about the value of these two patterns. The second is that these two patterns also collected relatively large proportions of the free-text comments. The basic criteria that we used for categorising the issues raised in these were: participants' experiences; software quality issues; and object-oriented aspects. We coded each response using a limited set of words for each of these categories, deriving the words from the comments themselves.

7.1. 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.*" [1]

Twenty participants who rated this pattern among their 'top three' provided comments, and eight who rated it under their 'not desirable' choices provided comments. The positive aspects were seen as being that its use makes a system extensible and maintainable, as well as aiding decomposition and reducing coupling. However, its abstraction and relative complexity meant that it was not easily understood and could easily be misused, leading to structures that were hard to maintain, making it unsuitable for use by novices. Few responses really gave any details about experiences in support of their views, but two examples of positive views that did so were:

- "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." [Developer]
- "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." [Researcher]

The similarity between the examples above is quite striking. On the less positive side, with a strong emphasis upon issues relating to coding and maintenance:

- "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." [Researcher]
- "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." [Developer]
- "To avoid procedural dependencies of conditionals, I prefer to use idioms that utilise 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 do not distract from the focus of the role being designed, and provide less opportunity for defects to be injected into the system." [Developer]

(The reference to "awareness of handshaking" concerns the need for the *Visitor* to consult all of the objects in the composite structure by sending requests to each concrete object and awaiting their responses.)

7.2. The singleton pattern

> "*Ensure a class only has one instance, and provide a global point of access to it.*" [1]

For this pattern we had 19 comments from those who rated it highly and eight who recorded having negative experiences with its use. This pattern does appear to be controversial, even one of the *GoF* authors (Erich Gamma) has said that he was in favour of dropping *Singleton*.[2] While it is valued for the benefits arising from duplication of similar objects, survey participants considered it to be massively misused and overused to provide global variables in a system.

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)." [Researcher]
- "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." [Researcher]

While the opposing views included:

- "Singleton is more an anti-pattern and introduces global state." [Developer]
- "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. [Developer]

8. Discussion

We begin by considering possible threats to the validity of our results and then compare the outcomes with those from some other relevant studies, including one of our own. We then consider the implications of the outcomes for research, practice and teaching.

8.1. Threats to validity

For a survey, there are three elements that need to be considered as potential sources of bias: the survey instrument; the process of administration of the survey; and the analysis of the data collected. We therefore examine each of these in turn.

8.1.1. Design of the survey instrument

Kitchenham and Pfleeger identify a number of validity issues that can arise for a survey instrument [26]. Of these, the most relevant is that of *content validity*, which is a subjective assessment of how appropriate the instrument seems to a group of reviewers with knowledge of the subject matter. We did ask two experienced external reviewers to assess our questionnaire, using a 9-question evaluation form for their responses. They identified a number of issues with the design, mainly about clarity and presentation, and these were duly addressed. We were unable to identify any similar surveys that could have been used to help assess our questionnaire for *criterion validity* [26].

In terms of *internal validity*, the question ordering used was for demographic data, rating of patterns, and ranking of patterns. We sought to reduce the likelihood of bias arising from the rating question (having the choices for the subsequent ranking question influenced by the order in which patterns were presented and assessed for rating) by using a matrix format. This ensured that all elements of the rating question were on the screen at the same time so that respondents were free to enter their ratings in whatever order they preferred. However, assuming that many would probably rate the patterns in the order provided, this might have created a 'rating fatigue' effect that could have affect the responses for the later patterns.

8.1.2. Sampling

Our *target population* was the set of all software designers and maintainers who had some form of experience with the use (or misuse) of design patterns. A key question is therefore whether the *actual population* that we sampled (all identifiable authors of papers about patterns) can be considered to constitute a valid *representative subset* of that *target* population [27]. In addition, there is the question as to whether the number of responses was large enough to permit analytical generalisation.

One problem here is that we have no means of identifying any characteristic of the target population (size, educational profile, etc.) that could be used to check our actual population. Characteristics of this population that might make us question its representativeness are:

- the level of education is high (see Table 3);
- the sample includes a large proportion of people who have written patterns (Table 8);
- the three groups were not truly random, due to the sampling mechanisms that we employed.

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

These factors do not invalidate our results but they do indicate that the results might not be fully representative of the target population, and hence that we cannot be sure of the *external validity* of the outcomes. However, as over a third of those responding were currently working as developers, their involvement does provide some element of validation for the outcomes.

The size of the actual population meant that we did not need to *sample*, we simply sent requests to all authors that we could identify. We have no reason to suggest that the non-responses from those addresses that were no longer valid biased the resulting set of respondents in any way.

Determining whether our sample size allows a degree of analytical generalisation is obviously difficult, given that we do not know the size or profile of the actual population. However, what we can reasonably infer is a description of the type of developer for which our results are relevant: namely someone likely to have a higher degree, possessing several years of experience with design patterns, and also likely to have had some experience of research.

8.1.3. Analysis

Kitchenham and Pfleeger identify three areas where analysis can affect the validity of the outcomes [28].

The first is that of *data validation*, and concerns the consistency with which data has been vetted. As indicated earlier, we received 227 responses and removed 21 of these on the basis of their being incomplete. No other filtering was performed and hence we would assess this aspect as being a very minor threat to internal validity.

The second potential threat is that of *data coding*. Although we performed no formal coding of the raw quantitative data, there was one minor coding element in our processing of this. This was that we reduced the four-point Likert scale to what was effectively a two-point scale when presenting the outcomes of the rating question. As our argument for having an even number of points on the scale was to force a choice, we consider that this reduction in scale was unlikely to have been significant.

The third element is that of the analysis itself. We consider the main shortcoming in analysis of the quantitative data to be the lack of any checks on consistency of answers. Where multiple rating questions that address the same concept are used, it is possible to employ a form such as Cronbach's alpha to assess reliability [31]. Unfortunately this cannot be used with a mix of forms such as we used here, and so our analysis lacked any reliability factor. For the qualitative element, the chief factor is the relatively limited number of useful responses received. Since our comments are based upon quite small numbers of responses, these should not be given too much weight.

8.2. Comparison with the mapping study

In comparing the outcomes with the findings from our previous study that acted as a motivation for undertaking this survey [10], we have looked at three aspects: the choice of patterns used in the empirical studies; the patterns where the empirical studies identified potentially conflicting outcomes; and the context within which the different studies have been conducted.

8.2.1. Choice of patterns

One of our concerns about the experimental studies was that the choice of patterns used was not explained in any of the papers. However, the three patterns studied most extensively in these were *Composite*, *Observer* and *Visitor*, all of which do appear in Table 19 and hence were clearly appropriate choices. Our survey therefore helps to provide greater confidence in the relevance of the mapping study outcomes.

8.2.2. Results from the empirical studies

In [10], we did discuss the qualitative data related to the three patterns that were most widely studied, because this data provided a useful interpretation of the outcomes. Here, we examine how this data relates to the outcomes from our survey for each pattern concerned.

- For *Observer* the only risk associated with its use that was identified from the mapping study was that of producing overly-complicated structures (noted by two of the primary studies). This agrees with the generally positive view of this pattern that emerges from our survey. Many respondents tended to emphasise its value as providing the means of addressing a frequently-encountered problem and also in reducing coupling (although one respondent felt that this was increased by its use). Another respondent, while positive about it, did caution that "major problems occur if observers react to changes to changing (other) models and one can run into cascades or infinite loops", which again agrees well with the concern about complexity in the mapping study outcomes.
- The *Composite* pattern seems to have provoked little controversy or disagreement in either study. In the mapping study, one primary study did note the need to understand recursion for successful use, but no other issues were recorded. From the survey, there were many positive views expressed, such as: "the most crucial pattern"; "can be seen in almost all applications"; and "makes it easy to create generic data models". Many respondents mentioned extensibility and flexibility. The only dissent came in two comments which, rather than questioning its value, questioned its right to be termed a pattern ("simple enough to not quality as a pattern any more" and "hardly a pattern").
- Both studies drew ambivalent reactions about *Visitor*. Several primary studies implied that successful understanding of its working required good documentation, and as we have seen in the earlier discussion, one survey respondent raised a similar point.

Overall, while the qualitative data was limited for both studies, it did tend to be in agreement for all three patterns.

8.2.3. Context of the studies

Here we do find a marked distinction between the material of our survey and that of the mapping study. As indicated in Tables 9 and 10, the respondents in this survey had mainly used patterns for software *development*. However, the experimental studies were largely concerned with making changes to systems that contained patterns, and hence were strongly focused upon *maintenance* activities. In addition, some of the observational papers also reported about maintenance activities, most particularly the very thorough analysis provided in [21].

We consider this to be an important distinction, and one that limits how far it is reasonable to use our survey to interpret and explain the outcomes of the mapping study. For the survey, the participants had mostly gained their experience of patterns from *development* (and, due to our sampling mechanism, most were also likely to be enthusiasts about patterns). For the empirical studies, the participants were performing *modification* tasks, and in the absence of other information, should be considered as probably somewhat agnostic about patterns.

Even so, there do seem to be some shared issues, particularly where the use of *Visitor* is concerned.

8.3. Comparison with other studies

Our mapping study included papers published up to the end of 2007. Within this period, we did identify one relatively informal

survey by Khomh and Guéhéneuc, initially reported in [32], and subsequently in [33]. This reported the experiences of 20 respondents and their assessment of the impact of the *GoF* patterns upon ten quality attributes, including the three highlighted in the *GoF*: reusability, expandability and understandability.

This survey used a set of rating questions, based on a five-point Likert Scale, together with a 'not applicable' option. For analysis, they merged the two positive and two negative points, to produce a reduced 3-point scale.

Their papers report in detail upon three patterns. For *Composite*, the assessments for the different attributes were either positive or neutral, which largely agrees with the views that we see in this survey. In the case of *Abstract Factory* the results were a mix of positive and negative, while for *Flyweight* all but one of the attributes were assessed negatively, which again broadly agrees with the perceptions that we have recorded.

Their survey was a much smaller sample than ours, and while we have some methodological reservations about the sampling (which is not reported in any detail), their focus upon specific qualities is a valuable contribution. While there is limited scope for direct comparison with our results, their assessment of the 23 patterns against the three attributes above (Table 2 of [33]) may help to explain some of the ratings we observe in Table 17. For each pattern, they assess whether it made a positive or negative contribution to that attribute when used for designing software. Two patterns had a negative effect on all three attributes: *Flyweight* and *Memento*, both of which had very high 'no experience' ratings in our survey as well as being ranked as 'not useful'. At the other end, four patterns had positive effects upon all three attributes: *Prototype*, *Composite*, *Interpreter* and *Iterator*. Although only *Composite* was highly ranked in our survey, both this pattern and *Iterator* had very positive scores in our rating question (the most directly comparable one), while *Interpreter* and *Prototype* were less well known as well as less well rated. So although the two surveys address different qualities, and are reported differently, they do appear to exhibit similar trends, particularly at the more 'negative' end of assessment.

8.4. Observations from the survey

A very simple summary of the outcomes would be to the effect that only approximately one quarter of the patterns described in the *GoF* are widely considered to be useful—although even then there appear to be caveats about the use of *Visitor* and *Singleton*. However, we do need to recognise that some patterns do also address issues that will only rarely be encountered in every application domain (e.g. *Proxy*, *Chain of Responsibility*) and so are less likely to be selected in our ranking questions.

Equally, around one quarter of the patterns are clearly considered to be of little use (or are very little known), and in the case of *Flyweight*, its use seems to be considered to be positively disadvantageous.

In Table 23 we seek to summarise our outcomes as a set of rather general recommendations about the use or avoidance of particular patterns during software development. Note too that ultimately, all decisions about use must depend upon the individual situation, and that all of the 23 patterns did get some 'very useful' ratings. For each entry, we have indicated the basis for our categorisation in terms of the different elements of the survey. Not all patterns from the *GoF* are included, as the rating and ranking assessments for many were not sufficiently in agreement for us to be able to make recommendations.

For researchers, this survey presents some outstanding questions (particularly about the links between development and maintenance), and also some pointers as to where future research might be most usefully directed.

Indeed, when considering future research, one other point that we should note is that most of our respondents have given their views as software *developers* (regardless of their main role), whereas other studies of patterns (such as the experimental studies) are much more focused upon software *maintenance*. As the study in [33] illustrates, the assessment of a pattern's usefulness can vary significantly according to which of these viewpoints is used.

8.5. Observations upon our survey design

As a final element, we offer some thoughts about how surveys such as this should be structured. Conducting a survey is a time-consuming exercise and unlike experiments, it is difficult to re-run a survey in a domain such as software engineering, where the sampling frame is likely to be small. Here we identify two aspects of our design choices that might usefully have been different. These points mainly expand slightly on the issues identified as being threats to validity.

The first change would be to have used multiple versions of the survey, with each using a different ordering of the patterns in the rating question. A set of four different orderings would have been sufficient for us to check for possible 'rating fatigue' in our respondents. Our reason for using a single ordering was that we expected only a few responses and hence assumed that there would be little opportunity to check for this effect. However, there would have been no disadvantage to designing our survey to cope with a larger number of responses.

The second would be to use a more simple structure for the questions. In particular, the use of multiple forms of rating

Table 23
Recommendations drawn from our results and observations.

Category	Patterns	Rating data	Ranking data/comments
Useful	Abstract Factory	Strong support, few negative ratings	In the top six choices
	Composite	Strong support, few negative ratings	In the top six choices
	Facade	Strong support, few negative ratings	In the top six choices
	Factory Method	Strong support, few negative ratings	No negative votes
	Observer	Strong support, few negative ratings	Top choice
	Iterator	Strong support, few negative ratings	Largely positive votes
Use with care	Singleton	Good support, but some strongly negative ratings	Very mixed views expressed, largely due to potential for misuse
	Visitor	Good support, but a large proportion of 'useful' ratings	Useful but complex
Better avoided	Flyweight	Few strong positive ratings	In the most negative four choices
	Interpreter	Few strong positive ratings	Mostly negative votes
	Memento	Unfamiliar to many, few positive ratings	In most negative four choices with no positive votes at all
	Prototype	Few strong positive ratings	Developers votes all negative

question, rather than the mix of rating and ranking questions we adopted, would have allowed us to check for reliability in the responses as well as presenting respondents with a simpler, more consistent, structure. The reason for including the ranking question was primarily to aid respondents with providing causal reasoning about pattern preferences. While it did achieve this, few responses were sufficiently well-formed to be of significant benefit.

9. Conclusions

Our survey has generated clear quantitative answers to the question: "which design patterns from the *GoF* do expert pattern users consider as useful or not useful for software development and maintenance, and why?". However, answering the qualitative coda ("and why?") has proved more problematical. Many of the issues identified will probably be familiar to the patterns community, at least informally, but one value of a survey is that it organises the knowledge elicitation in a systematic and rigorous manner.

The concept of a 'knowledge schema' is clearly a familiar one in design [14], but a schema is generally personal to a designer, so that the knowledge being reused is their own. Software design patterns seek to make the schemas of experienced designers available for others to use. The results of our survey suggest that this is not universally successful, although it clearly works well for some patterns. There is also a related question as to how far patterns can be employed by relative novices [34].

Most of the respondents to our survey were certainly not novices, with well over half of them having more than 10 years of experience with developing OO systems. It is therefore interesting that, given this wealth of experience, only three patterns really seem to be highly regarded, with few caveats being made about their use (*Observer*, *Composite* and *Abstract Factory*). We have subsequently investigated further as to why we also received such differing views about *Visitor*, *Singleton* and *Facade*, with our findings reported in [35].

In Section 8.4 we have identified some issues and recommendations for practice and research (and hence, indirectly, for teaching too), that stem from these findings. As software engineering practices move towards fuller use of an 'evidence base', one of the needs is to establish what works, when and where. One value of a survey is that it can help to identify the scope of individual patterns, and although this survey does not provide deep understanding of the mechanisms (the 'why'), it does provide a valuable step towards identifying what further questions should be asked. In particular, it highlights the point that not all patterns are equally useful and hence that generalisations about their use are best avoided.

Acknowledgements

We would like to thank all those who participated in our survey, and especially those who provided us with interpretations of their views. We would also like to thank Prof. Barbara Kitchenham and Prof. Ray Welland for acting as external reviewers for our survey, and Prof. Kitchenham for her advice on the statistical analysis.

References

[1] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995.
[2] B. Kitchenham, T. Dybå, M. Jørgensen, Evidence-based software engineering, in: Proceedings of ICSE 2004, IEEE Computer Society Press, 2004, pp. 273–281.
[3] D. Budgen, J. Bailey, M. Turner, B. Kitchenham, P. Brereton, S. Charters, Cross-domain investigation of empirical practices, IET Software 3 (2009) 410–421. EASE special section.
[4] B. Kitchenham, P. Brereton, D. Budgen, M. Turner, J. Bailey, S. Linkman, Systematic literature reviews in software engineering – a systematic literature review, Information and Software Technology 51 (2009) 7–15.
[5] B. Kitchenham, R. Pretorius, D. Budgen, P. Brereton, M. Turner, M. Niazi, S. Linkman, Systematic literature reviews in software engineering – a tertiary study, Information and Software Technology 52 (2010) 792–805.
[6] F.Q. da Silva, A.L. Santos, S. Soares, A.C.C. França, C.V. Monteiro, F.F. Maciel, Six years of systematic literature reviews in software engineering: an updated tertiary study, Information and Software Technology 53 (2011) 899–913.
[7] M. Jørgensen, Evidence-based guidelines for assessment of software costs uncertainty, IEEE Transactions on Software Engineering 31 (2005) 942–954.
[8] J. Hannay, T. Dybå, E. Arisholm, D. Sjøberg, The effectiveness of pair programming. a meta analysis, Information and Software Technology 51 (2009) 1110–1122.
[9] D. Budgen, A. Burn, P. Brereton, B. Kitchenham, R. Pretorius, Empirical evidence about the UML: a systematic literature review, Software – Practice and Experience 41 (2011) 363–392.
[10] C. Zhang, D. Budgen, What do we know about the effectiveness of software design patterns?, IEEE Transactions on Software Engineering 38 (2012) 1213–1231.
[11] B.A. Kitchenham, D. Budgen, O.P. Brereton, Using mapping studies as the basis for further research – a participant–observer case study, Information and Software Technology 53 (2011) 638–651. Special section from EASE 2010.
[12] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, S. Angel, A Pattern Language, Oxford University Press, 1977.
[13] B. Adelson, E. Soloway, The role of domain experience in software design, IEEE Transactions on Software Engineering 11 (1985) 1351–1360.
[14] F. Détienne, Software Design – Cognitive Aspects, Springer Practitioner Series, 2002.
[15] C. Kohls, K. Scheiter, The relation between design patterns and schema theory, in: Proceedings of the 15th Conference on Pattern Languages of Programs (PLoP'08), ACM Press, 2008, pp. 1–14.
[16] D.S. Cruzes, T. Dybå, Research synthesis in software engineering: a tertiary study, Information and Software Technology 53 (2011) 440–455.
[17] W. Shadish, T. Cook, D. Campbell, Experimental and Quasi-Experimental Design for Generalized Causal Inference, Houghton Mifflin Co., 2002.
[18] T. Dybå, T. Dingsøyr, Empirical studies of agile software development: a systematic review, Information and Software Technology 50 (2008) 833–859.
[19] P. Mohagheghi, V. Dehlen, Where is the proof? – A review of experiences from applying MDE in industry, in: ECMDA-FA, LNCS, vol. 5095, Springer, 2008, pp. 432–443.
[20] D. Budgen, C. Zhang, Preliminary reporting guidelines for experience papers, in: Proceedings of EASE 2009, BCS eWiCs, 2009, pp. 1–10.
[21] P. Wendorff, 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), IEEE Computer Society Press, 2001, pp. 77–84.
[22] A. Fink, The survey handbook, The Survey Kit, second ed., vol. 1, Sage Books, 2003.
[23] B.A. Kitchenham, S.L. Pfleeger, Principles of survey research part 2: designing a survey, ACM Software Engineering Notes 21 (2002) 18–20. For Part 1, see under Pfleeger.
[24] S.L. Pfleeger, B.A. Kitchenham, Principles of survey research part 1: turning lemons into lemonade, ACM Software Engineering Notes 26 (2001) 16–18.
[25] B.A. Kitchenham, S.L. Pfleeger, Principles of survey research part 3: constructing a survey instrument, ACM Software Engineering Notes 27 (2002) 20–24.
[26] B.A. Kitchenham, S.L. Pfleeger, Principles of survey research part 4: questionnaire evaluation, ACM Software Engineering Notes 27 (2002) 20–23.
[27] B.A. Kitchenham, S.L. Pfleeger, Principles of survey research part 5: populations and samples, ACM Software Engineering Notes 27 (2002) 17–20.
[28] B.A. Kitchenham, S.L. Pfleeger, Principles of survey research part 6: data analysis, ACM Software Engineering Notes 28 (2003) 24–27.
[29] D.A. de Vaus, Surveys in Social Research, fifth ed., Routledge, 2002.
[30] A. Field, Discovering Statistics using SPSS, third ed., Sage Publications Ltd., 2009.
[31] L.J. Cronbach, Coefficient alpha and the internal structure of tests, Psychometrika 16 (1951) 297–334.
[32] F. Khomh, Y.-G. Guéhéneuc, Perception and reality: what are design patterns good for?, in: Proceedings of 11th ECOOP Workshop on Quantitative Approaches in Object Oriented Software Engineering (QAOOSE), Springer-Verlag, 2007, p 7.
[33] F. Khomh, Y.-G. Guéhéneuc, Do design patterns impact software quality positively? in: Proceedings of CSMR 2008, pp. 274–278.
[34] I. Sommerville, Software Engineering, eighth ed., Addison-Wesley, 2007.
[35] C. Zhang, D. Budgen, S. Drummond, Using a follow-on survey to investigate why use of the visitor, singleton and facade design patterns is controversial?, in: Proceedings 6th International Symposium on Empirical Software Engineering and Measurement (ESEM), ACM Press, 2012, pp. 79–88.