Perception and Reality: What are Design Patterns Good For?

Foutse Khomh
Yann-Gaël Guéhéneuc
Ptidej Team, GEODES, DIRO, University of Montreal, Canada
{foutsekh,guehene}@iro.umontreal.ca

Abstract

We present a study of the impact of design patterns on quality attributes. An empirical study is performed by asking respondents their evaluations of the impact of all design patterns on several quality attributes. We present detailed results for three design patterns (Abstract Factory, Composite, and Flyweight) and three quality attributes (reusability, understandability, and expendability). We perform a Null hypothesis test and we conclude that, contrary to popular beliefs, design patterns do not always improve reusability and understandability, but that they do improve expandability.

1 Introduction

Many studies in the literature present design patterns as a promising solutions to improve the quality of object oriented software systems during development. It is widely claimed that they improve the quality of systems and that every well-structured object oriented architectures contain patterns [3].

However some studies suggested that the use of design patterns do not always results in good quality design. In particular, a tangled implementation of these patterns in a design impacts negatively the quality that these patterns claimed to improve [4]. Also design patterns generally increase the complexity of an initial design to ease future enhancements.

Thus, to the best of our knowledge, evidence of quality improvements through the use of design patterns consists primarily of intuitive statements and examples. There is little empirical evidence to support the claims of improved flexibility, reusability, adaptability as put forward in [3] when applying design patterns. Also, the impact of design patterns on other quality attributes is unclear.

This lack of evidence around the benefits of design patterns and their impact on design quality led us to carry out an empirical study on the impact of design patterns on the quality of systems as perceived by software developers and with respect to known principles of the object oriented paradigm.

In this work, we present a survey carried out over a population of experienced object-oriented developers and its results to attempt answering the question: *is the impact of design patterns on quality attributes positive, neutral, or negative?* We conclude by a discussion on the results.

2 Related Work

Since the introduction of design patterns by Gamma et al. [3], there has been a growing interest on the use of design patterns, many work have been carried out to study the potential impacts of this concept on software systems but very few investigated empirically the impact on quality. We present here only examples of the main work on design patterns.

Wydaeghe et al. [6] presented a study on the concrete use of six design patterns when building an OMT editor. They discussed the impact of these patterns on quality attributes such as reusability, modularity, flexibility, and understandability. They also discuss the difficulty of the concrete implementation of these patterns. They concluded that although design patterns offer a lot of advantages, not all patterns have the same effects on the quality attributes. However, this study is limited to the authors' own experience and thus their appreciation of the impact of these patterns on quality can hardly been generalized to any context of development.

Tahvildari et al. [5] studied the 23 design patterns from [3] and presented a layered classification of the primary relationships between these patterns: use, refine, and conflict, and three secondary relationships: similar, combine, and require (that can be expressed in terms of the primary ones). They organized the design patterns into two abstraction levels. They dis-

cussed how their classification can assist software engineers with understanding better the complex relationships between patterns, organizing existing patterns as well as categorizing and describing new patterns and building tools that support the application of patterns during restructuring. However, they did not investigate whether the use of these patterns really improve the quality of designs.

McNatt and Bieman [4] examined the coupling between design patterns. They dressed a parallel between modularity and abstraction in software systems and modularity and abstraction in patterns. They concluded that when patterns are loosely coupled and abstracted then maintainability, factorability, and reusability are well supported by the patterns. They also concluded on the need for further studies to understand effective pattern constructs and good pattern coupling methods.

Bieman et al. [2, 1] examined common recommended programming styles on several different software systems, with and without patterns, and concluded that in contrast with common claims the use of design patterns can lead to more change prone classes rather than less change prone classes during the evolution of the systems.

3 Problem Formulation

There are little evidence on the impact of design patterns on the quality of software systems. Most of the statements supporting the hypothesis of improvements of the quality are intuitive.

This work aims at quantifying the impact of design patterns on the overall quality of systems. We had the choice between an absolute, a relative, or an empirical quantification.

Due to the lack of a well defined framework for the evaluation of the quality of systems, we chose an empirical quantification consisting of collecting and analyzing evaluations by software developers of certain aspects of the quality of systems that design patterns may impact.

4 Method

We built a questionnaire and carried out a survey electronically during the period of January to May 2007.

4.1 Our Questionnaire

We chose, based on their relevance to design patterns and software systems, the following golden set of quality attributes:

- Related to architecture and design:
  - **Expandability:** The degree to which architectural, data, or procedural design can be extended.
  - **Simplicity:** The degree to which the architecture of the system can be understood without difficulty.
  - **Reusability:** The degree to which a piece of design (or a subset of a piece of design) can be reused in another design.
- Related to implementation:
  - **Learnability:** The degree to which the code source of a system is easy to learn by new developers.
  - **Understandability:** The degree to which the code source of the system can be understood without difficulty.
  - **Modularity:** The degree to which the implementation of functions in a system are independent from one another.
- Related to runtime:
  - **Generality:** The degree to which a system can perform a wide range of functions at runtime.
  - **Modularity at runtime:** The degree to which functions of a system are independent from one another at runtime.
  - **Scalability:** The degree to which the system can cope with large amount of data and computation at runtime.
  - **Robustness:** The degree to which a system continues to function properly under abnormal conditions or circumstances.

Each quality attribute was evaluated using a six-point Likert scale:

```
A - Very positive
B - Positive
C - Not significant
D - Negative
E - Very Negative
F - I don't know
```

For every design pattern in [3] and for every quality attribute from our golden set, the respondents were

asked to assess the impact of the pattern on the quality of a system in which the pattern would have been used appropriately. For example, for the Composite design pattern and Learnability, the respondents were asked to assess the impact of the pattern on the overall Learnability of a system implementing the pattern appropriately.

4.2 Data Collection

The questionnaire was sent to experienced object-oriented developers around the world and posted on three specialized mailing lists, refactoring, patterns-discussion, and gang-of-4-patterns.

We asked the respondents to consider the situation where patterns where used appropriately in programs to solve their corresponding design problems.

Among the answers that we received, we selected the questionnaires of 20 developers with a long experience in the use of design patterns in software development.

Among the selected 20 questionnaires, some respondents did not evaluate the quality of all design patterns. Thus, some patterns have more evaluation than others.

4.3 Data Processing

To answer our question: *is the impact of design patterns on quality attributes positive, neutral, or negative?* and due to the high level of variations between answers, we chose to aggregate answers `A` and `B` and answers `D` and `E`:

```
Positive = A and B
Neutral  = C
Negative = D and E
```

Answers `F` were not considered to assess the impact of design patterns on quality because we considered that the respondent did not know how the pattern impacts the quality attribute.

Using the previous three-point Likert scale, we computed the frequencies of the answers on each quality attribute: Positive, Neutral, and Negative and we carried out a Null test to decide on the impact of the patterns on the quality attributes according to the respondents.

5 Results of the Survey

We present in the following the results for three design patterns: Abstract Factory, Composite, and Flyweight, and the three quality attributes related to design patterns: reusability, expandability, and understandability. Results for all design patterns and quality attributes will be presented in a future work.

Attributes	Positive	Neutral	Negative
Expandability	100.0	0.0	0.0
Simplicity	69.23	15.38	15.38
Generality	76.92	15.38	7.69
Modularity	71.43	21.43	7.14
Modularity at Runtime	53.85	38.46	7.69
Learnability	76.92	7.69	15.38
Understand-ability	69.23	15.38	15.38
Reusability	61.54	23.08	15.38
Scalability	41.67	41.67	16.67
Robustness	8.33	91.67	0.0

Table 1. Impact of Composite on quality attributes.

5.1 Qualitative Analysis

5.1.1 Design Patterns

We chose the following three design patterns to illustrate the opinions of our respondents about the impact of design patterns on quality attributes firstly because of their popularity, they are among commonly used patterns thus we felt that their evaluation would be more accurate, and secondly because they appeared to be considered by our respondents as globally positive (Composite), globally neutral (Abstract Factory), and globally negative (Flyweight). Therefore, we felt that they would be more representative.

Composite. Table 1 presents the evaluations by the respondents of the impact of the Composite pattern on the quality attributes. Looking at the table, it appears that the Composite pattern is mostly perceived as having a positive impact on the quality of systems. All quality attributes are impacted positively but for the scalability and robustness that are not positive. Given the purpose of the Composite pattern, having a neutral impact on scalability is rather surprising.

Abstract Factory. Table 2 presents the evaluations by the respondents of the impact of the Abstract Factory pattern on the quality attributes. The table shows that half the quality attributes is considered as positively impacted while the other half is not. It is not surprising that the pattern is overall judged as neutral given its purpose and complexity. However, it is striking that both learnability and understandability are felt negatively impacted.

Attributes	Positive	Neutral	Negative
Expandability	100.0	0.0	0.0
Simplicity	53.33	13.33	33.33
Generality	78.57	21.43	0.0
Modularity	85.71	7.14	7.14
Modularity at Runtime	46.15	38.46	15.38
Learnability	35.71	28.57	35.71
Understand-ability	38.46	30.77	30.77
Reusability	50.0	42.86	7.14
Scalability	21.43	64.29	14.29
Robustness	0.0	72.73	27.27

Table 2. Impact of Abstract Factory on quality attributes.

Attributes	Positive	Neutral	Negative
Expandability	22.22	44.44	33.33
Simplicity	0.0	22.22	77.78
Generality	11.11	44.44	44.44
Modularity	33.33	33.33	33.33
Modularity at Runtime	11.11	66.67	22.22
Learnability	0.0	20.0	80.0
Understand-ability	0.0	10.0	90.0
Reusability	37.5	12.5	50.0
Scalability	77.78	0.0	22.22
Robustness	22.22	66.67	11.11

Table 3. Impact of Flyweight on quality attributes.

Flyweight. Table 3 presents the evaluations by the respondents of the impact of the Flyweight pattern on the quality attributes. The table reports that this patterns is perceived as impacting negatively all quality attributes but for the scalability. Given the purpose of the pattern, it is not surprising that its impact on scalability is judged positively. The negative perception could be explained by the less frequent use of Flyweight in comparison with Composite and Abstract Factory.

5.1.2 Quality Attributes

We chose the following three quality attributes because it is claimed in [3] that they are improved by the use of design patterns.

Patterns	Positive	Neutral	Negative
A.Factory	46.15	46.15	7.69
Builder	36.36	45.45	18.18
F.Method	60.0	20.0	20.0
Prototype	63.64	0.0	36.36
Singleton	18.18	54.55	27.27
Adapter	66.67	25.0	8.33
Bridge	41.67	16.67	41.67
Composite	58.33	25.0	16.67
Decorator	36.36	18.18	45.45
Facade	36.36	45.45	18.18
Flyweight	37.5	12.5	50.0
Proxy	45.45	36.36	18.18
Ch.Of.Resp	54.55	27.27	18.18
Command	30.0	20.0	50.0
Interpreter	50.0	0.0	50.0
Iterator	72.73	9.09	18.18
Mediator	20.0	50.0	30.0
Memento	28.57	42.86	28.57
Observer	53.85	23.08	23.08
State	20.0	40.0	40.0
Strategy	41.67	33.33	25.0
T.Method	58.33	33.33	8.33
Visitor	28.57	28.57	42.86

Table 4. Impact of design patterns on reusability.

Reusability. Table 4 presents the evaluations by respondents of the impact of design patterns on reusability. Overall, as shown in Table 8, reusability is felt as being slightly more negatively impacted by design patterns, with 12 negative patterns and 11 positive patterns. This is rather surprising as the use of design patterns is claimed to improve reusability according to the GoF.

Expandability. Table 5 presents the evaluations by the respondents of the impact of design patterns on the expendability. All respondents felt that expendability is improved when using design patterns, in conformance with what is expected of using patterns.

Understandability. Table 6 presents the evaluations by the respondents of the impact of design patterns on the understandability. Similarly to reusability, respondents felt that the understandability was rather slightly negatively impacted by the use of patterns.

Patterns	Positive	Neutral	Negative
A.Factory	100.0	0.0	0.0
Builder	90.91	9.09	0.0
F.Method	72.73	9.09	18.18
Prototype	63.64	27.27	9.09
Singleton	9.09	27.27	63.64
Adapter	50.0	41.67	8.33
Bridge	83.33	16.67	0.0
Composite	100.0	0.0	0.0
Decorator	90.91	0.0	9.09
Facade	58.33	16.67	25.0
Flyweight	22.22	44.44	33.33
Proxy	45.45	45.45	9.09
Ch.Of.Resp	91.67	8.33	0.0
Command	66.67	16.67	16.67
Interpreter	63.64	27.27	9.09
Iterator	90.91	9.09	0.0
Mediator	58.33	25.0	16.67
Memento	33.33	55.56	11.11
Observer	85.71	7.14	7.14
State	72.73	18.18	9.09
Strategy	76.92	15.38	7.69
T.Method	84.62	15.38	0.0
Visitor	71.43	7.14	21.43

Table 5. Impact of design patterns on expandability.

Patterns	Positive	Neutral	Negative
A.Factory	38.46	30.77	30.77
Builder	81.82	9.09	9.09
F.Method	45.45	27.27	27.27
Prototype	58.33	16.67	25.0
Singleton	91.67	8.33	0.0
Adapter	50.0	25.0	25.0
Bridge	50.0	33.33	16.67
Composite	75.0	16.67	8.33
Decorator	45.45	9.09	45.45
Facade	81.82	18.18	0.0
Flyweight	0.0	10.0	90.0
Proxy	33.33	50.0	16.67
Ch.Of.Resp	33.33	33.33	33.33
Command	33.33	33.33	33.33
Interpreter	63.64	0.0	36.36
Iterator	50.0	41.67	8.33
Mediator	58.33	25.0	16.67
Memento	33.33	55.56	11.11
Observer	42.86	35.71	21.43
State	54.55	0.0	45.45
Strategy	69.23	23.08	7.69
T.Method	38.46	38.46	23.08
Visitor	21.43	21.43	57.14

Table 6. Impact of design patterns on understandability.

5.2 Quantitative Analysis

Using the results obtained by aggregating the previous data in Tables 1, 2, 3, 4, 5, and 6, and we carried out a Null hypothesis test to quantify the impact of the design patterns on the quality attributes. We use the frequencies of Positive and non-positive (combined Neutral and Negative answers) to decide on the impact of a given pattern on a specific quality attribute.

For a given question about the impact of a pattern on a quality attribute, we considered the random variable X, that takes the value 0 when the impact of the pattern on the attribute is positive and 1 when the impact is not positive. We defined P as the probability that the pattern does not impact positively the attribute. The probability that the pattern impacts positively the attribute is therefore $1-P$. Considering the N respondents $j = 1, ..., N$ answering the question, we viewed their answers as occurrences of the random variable X and noted them: $X_1, X_2, ...X_N$. Then, we set our Null hypothesis to be H_0: The impact of the pattern on the quality attribute is positive, which yields, in terms of probability, to $P \leq \frac{1}{2}$. The alternative hypothesis is then H_1: The pattern does not impact positively the attribute, i.e., $P > \frac{1}{2}$. Hence, our decision rule is:

- We confirm H_0 if f_N is not high enough;
- We confirm H_1 if f_N is high enough.

where f_N is the frequency of the respondents who answered that the pattern impacts negatively or does not impact the attribute.

The risk we encountered by rejecting the Null hypothesis H_0, i.e, the pattern positively impacts the quality attribute, is then: $1-F(f_N)$, where F is the cumulative density of the Bernoulli distribution $\beta(N, \frac{1}{2})$.

The Null hypothesis test yields the results summarized in Tables 7 and 8. The analysis of the results of our survey revealed that in contrary to what is commonly admitted in the literature (which is that the use of design patterns yields to architectures that are reusable, simple and more understandable), the reality of the use of patterns is different. Developers consider that although patterns are useful to solve design problems, they do not always improve the quality of systems

in which they are used. Some patterns like Flyweight are even considered bad for the quality of systems. A large number of respondents consider that they sensibly decrease simplicity, learnability, and understandability.

6 Conclusion

In this paper, we presented a study of the impact of design patterns on quality attributes. This empirical study was performed by asking respondents their evaluations of the impact of all design patterns on several quality attributes. We concluded that, contrary to popular beliefs, design patterns do not always improve reusability and understandability, but that they do improve expendability.

However this study stands only on the opinion of a surveyed population of experienced developers and we cannot consider its results as free from uncertainty. In particular, some design patterns received no evaluations from some respondents because they are less known and used or, possibly, because they are judged up-front as impacting negatively quality as one respondent suggested. Moreover, the 20 respondents may not be representative of the general population of software developers.

In future work, we plan to carry out a wider survey by detailing our questionnaire and broadcasting it to more respondents. The questionnaire is available on the Internet at `http://www.iro.umontreal.ca/~ptidej/Questionnaire.pdf` or `http://ptidej.dyndns.org/downloads/` (it may take some minutes to load as it weighs 4 MB). We are looking forward receiving your kind contributions.

Acknowledgments

Foutse Khomh and Yann-Gaël Guéhéneuc were partially supported by NSERC, Discovery Grant.

We would also like to thank all the respondents of our questionnaire for their evaluations of the patterns and for all their very interesting comments.

References

[1] J. Bieman, G. Straw, H. Wang, P. W. Munger, and R. T. Alexander. Design patterns and change proneness: An examination of five evolving systems. In *Proceedings of the* 9^{th} *international Software Metrics Symposium*, pages 40–49. IEEE Computer Society Press, September 2003.

[2] J. M. Bieman, R. Alexander, P. W. M. III, and E. Meunier. Software design quality: Style and substance. In *Proceedings of the* 4^{th} *Workshop on Software Quality*. ACM Press, March 2001.

[3] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns – Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1^{st} edition, 1994.

[4] W. B. McNatt and J. M. Bieman. Coupling of design patterns: Common practices and their benefits. In *Proceedings of the* 25^{th} *Computer Software and Applications Conference*, pages 574–579. IEEE Computer Society Press, October 2001.

[5] L. Tahvildari and K. Kontogiannis. On the role of design patterns in quality-driven re-engineering. In *Proceedings of the*6^{th} *European Conference on Software Maintenance and Reengineering*, pages 230–240. IEEE Computer Society, March 2002.

[6] B. Wydaeghe, K. Verschaeve, B. Michiels, B. V. Damme, E. Arckens, and V. Jonckers. Building an OMT-editor using design patterns: An experience report. http://info.vub.ac.be/bwydaegh/tekst/paers/tools98/html/tools98.html, 1998.

Attributes	Composite		A.Factory		Flyweight	
	E	R(%)	E	R(%)	E	R(%)
Expendability	+	0.0	+	0.0	-	1.76
Simplicity	+	5.92	+	30.36	-	0.0
Generality	+	1.76	+	1.76	-	0.15
Modularity	+	5.92	+	0.37	-	5.92
Modularity at Runtime	+	30.36	-	30.36	-	0.15
Learnability	+	1.76	-	15.09	-	0.0
Understandability	+	5.92	-	15.09	-	0.0
Reusability	+	15.09	+	50.0	-	15.09
Scalability	-	30.36	-	1.76	+	1.76
Robustness	-	0.15	-	0.0	-	1.76
	8 + / 2 -		5 + / 5 -		1 + / 9 -	

Table 7. Estimation of the impact of the three design patterns on quality attributes.

Design Patterns	Expendability(%)		Understandability(%)		Reusability(%)	
	E	R(%)	E	R(%)	E	R(%)
A.Factory	+	0.0	-	15.09	+	50.0
Builder	+	0.15	+	0.37	-	15.09
F.Method	+	1.76	-	30.36	+	15.09
Prototype	+	30.36	+	30.36	+	30.36
Singleton	-	0.15	+	0.15	-	0.37
Adapter	+	30.36	-	30.36	+	5.92
Bridge	+	0.37	+	50.0	-	30.36
Composite	+	0.0	+	5.92	+	15.09
Decorator	+	0.15	-	30.36	-	5.92
Facade	+	30.36	+	1.76	-	5.92
Flyweight	-	1.76	-	0.0	-	15.09
Proxy	-	30.36	-	5.92	+	50.0
Ch.Of.Resp	+	0.15	-	5.92	+	30.36
Command	+	5.92	-	5.92	-	5.92
Interpreter	+	5.92	+	5.92	+	30.36
Iterator	+	0.15	+	50.0	+	5.92
Mediator	+	30.36	+	30.36	-	1.76
Memento	-	5.92	-	30.36	-	15.09
Observer	+	0.15	-	30.36	+	50.0
State	+	5.92	+	30.36	-	1.76
Strategy	+	1.76	+	15.09	-	30.36
T.Method	+	0.37	-	15.09	+	30.36
Visitor	+	5.92	-	1.76	-	1.76
	19 + / 4 -		11 + / 12 -		11 + / 12 -	

Table 8. Estimation of the impact of design patterns on the three quality attributes