2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)

How Are Design Patterns Concerned by Developers?

He Jiang*†, Dong Liu*, Xin Chen‡, Hui Liu†, and Hong Mei†
*School of Software, Dalian University of Technology, Dalian, China
Email: jianghe@dlut.edu.cn, dongliu05@gmail.com
†School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China
Email: liuhui08@bit.edu.cn, meihong@bit.edu.cn
‡School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China
Email: chenxin4391@hdu.edu.cn

***Abstract*—In recent years, design pattern has become an accepted concept in software design and many studies have involved various aspects of design patterns. However, it is an open question that how design patterns are discussed by developers.**

In this study, we conduct an empirical study to answer this question by soliciting Stack Overflow. First we build a new open catalog with 425 design patterns. Then, we extract 187,493 design pattern relevant posts from Stack Overflow.

As to these posts, we find that the popularity of design patterns follows a long tail distribution. More surprisingly, nearly half of the posts focus on only five design patterns. We also successfully detect many potential new co-occuring design patterns, which could well complement the deficiency of existing studies.

***Keywords*-design pattern; Stack Overflow; empirical study;**

I. Introduction

Design patterns in software engineering provide proven reusable solutions to commonly occurring problems by summarizing explicit good design practices. In the literature, a large number of studies have been conducted to investigate the issues around design patterns [1].

Despite these encouraging studies of design patterns, there remain many open questions to be investigated. *Have all design patterns drawn attention of developers? Does there exist a design pattern that can always keep attractive for developers? If not, what types of design patterns may gradually become outdated? Are there any related design patterns that are not introduced in the literature?* Obviously, the answers to the above questions are beneficial for both developers and researchers. On the one hand, developers could better acknowledge the emphases of mastering all the design patterns and gain more experience of how to correctly use these design patterns in practice. On the other hand, researchers could spend more efforts on clarifying these popular design patterns, e.g., providing more "known uses", code samples, and related design patterns.

To investigate the answers to these questions, we conduct empirical analysis by Stack Overflow, one popular online knowledge-sharing community which attracts millions of developers to share their experience [2].

II. Building Research Corpus

A. Building Design Pattern Catalog

The design patterns[1] are collected from three sources, including existing design pattern related books [3], e.g., GoF [4], POSA [5], PoEAA [6], online design pattern repositories, e.g., the Hillside repository[2], and the design pattern category of Wikipedia[3].

We merge all the design patterns from the above three sources together and remove the duplicates according to their names. Finally, there are 425 unique design patterns in our design pattern catalog[4].

B. Posts Extraction

First, we download the Stack Overflow question posts spanning from August 2008 to December 2017. Next, given a design pattern in the design pattern catalog, we regard a post as its relevant post candidate if the design pattern name appears in the title, body, or the tags of the post.

However, the design pattern relevant post candidates are still too noisy to use. For example, "visitor" can represent a design pattern of GoF, but it also refers to someone who visits a web page in some cases. Therefore, we use a Logistic Regression classifier to determine whether a post candidate of a design pattern is really relevant to the design pattern. The classifier is trained on 400 randomly selected and manually annotated post candidates with five extracted features[5].

Then, by applying the trained classifier to all the pairs of design patterns and their relevant post candidates, 196,485 pairs (involving 187,493 non-redundant posts and 210 design patterns) are predicted to be "relevant". Moreover, the results are validated manually on 385 randomly sampled pairs. With

[1]Please note that design patterns in this paper include both "design patterns" and "architectural patterns", as the definition of design pattern may vary in different sources.

[2]http://hillside.net/patterns

[3]https://en.wikipedia.org/wiki/Category:Software design patterns

[4]The design pattern catalog as well as some results of the findings are available on https://github.com/WoodenHeadoo/design-pattern-catalog/wiki

[5]Due to the limitation of space, the detailed description of the features are shown on https://github.com/WoodenHeadoo/design-pattern-catalog/wiki

2574-1934/19/$31.00 ©2019 IEEE
DOI 10.1109/ICSE-Companion.2019.00090

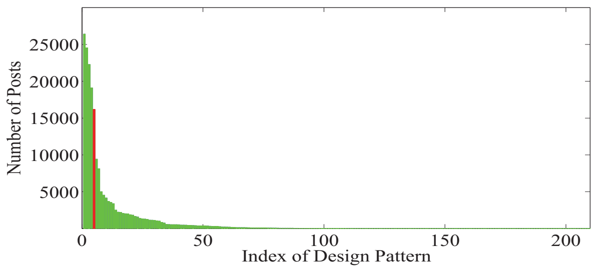

Figure 1: The number of relevant posts against each design pattern in descending order.

respect to the class of relevant, the values of precision and recall are 97.3% and 87.8%, respectively.

III. Findings

A. *Mostly Discussed Design Patterns*

The popularity of design patterns varies sharply. The 210 design patterns that can be found in Stack Overflow are sorted in descending order by their numbers of relevant posts (as shown in Fig. 1). From this figure, we can observe that only a small part of design patterns are frequently discussed by developers and the frequency distribution of design patterns exhibits a long tail. Moreover, almost half of the relevant posts are covered by the top 5 design patterns (the cutoff in the figure), namely *Model View Controller (MVC)*, *Active Record*, *Model View ViewModel (MVVM)*, *Reflection*, and *Dependency Injection (DI)*.

Both developers and researchers could benefit from this interesting finding (the long tail distribution of design patterns) by focusing on the most discussed design patterns.

B. *Evolvement of Design Patterns*

The popularity of design patterns gradually evolves over time. The evolvement trends of the design patterns initially undergo a strong rise with respect to the number of relevant posts. After reaching their peaks, the trends of design patterns related to traditional areas (e.g., database, fundamental programming) present a gradually downward trend, such as *Active Record*, *Reflection*, and *Singleton*. In contrast, the trends of some design patterns associated with emergent techniques still keep steadily upward, such as *Dependency Injection*, *Adapter*, *Future*, and *Pipeline*, which are relevant to *web application frameworks*, *Android development*, *concurrent programming*, and *data processing of machine learning*, respectively.

By investigating the applications of new technology waves, researchers may propose more influential design patterns.

C. *Co-occurrence of Design Patterns*

Two design patterns may share a same relevant post. To facilitate the observation, we present the 25 most frequently

Table I: Top 25 most frequent design pattern pairs

ID	Design Pattern Pair (# co-occurrences)	ID	Design Pattern Pair (# co-occurrences)
1	Unit of Work - Repository (545)	14	Factory Method - Factory (126)
2	Event Sourcing - CQRS (344)	15	MVVM - MVP (120)
3	Factory - DI (292)	16	Factory - Abstract Factory (112)
4	Repository - DI (278)	17	Unit of Work - DI (104)
5	MVVM - MVC (232)	18	Singleton - MVC (85)
6	Service Locator - DI (196)	19	MVP - DI (77)
7	Singleton - DI (194)	20	Strategy - Factory (76)
8	MVP - MVC (191)	21	Factory Method - Abstract Factory (73)
9	MVVM - DI (186)	22	Factory - Builder (71)
10	Singleton - Factory (163)	23	Service Layers - Repository (70)
11	Observer - MVC (162)	24	MVC - DAO (69)
12	Repository - MVC (151)	25	MVC - Active Record (69)
13	MVC - DI (127)		

co-occurring design pattern pairs in Table I. By an in-depth analysis on some of posts containing multiple design patterns, we observe that these posts are mainly about *cooperations* or *comparisons* of design patterns. That means they are potential related design patterns. In the table, most design pattern pairs have been provided in the literature, but some of them are rarely discussed explicitly in the literature, such as *Factory - DI* and *MVC - DAO*.

By referring to the potential related design patterns, developers could consider more relevant design patterns in resolving programming tasks and researchers could better enrich their design pattern related documents.

IV. Conclusion

Design patterns play an important role for developers in resolving some recurring problems. In this study, we conduct an empirical study to investigate the popularity, evolvement, and co-occurrence of design patterns in Stack Overflow.

With the results of the empirical study, developers could better acknowledge the emphases of design patterns and researchers could better acknowledge what design patterns should be further enriched about their documents.

References

[1] C. Zhang and D. Budgen, “What do we know about the effectiveness of software design patterns?” *IEEE Trans. Softw. Eng.*, vol. 38, no. 5, pp. 1213–1231, Sept 2012.

[2] L. Nie, H. Jiang, Z. Ren, Z. Sun, and X. Li, “Query expansion based on crowd knowledge for code search,” *IEEE Trans. Serv. Comput.*, vol. 9, no. 5, pp. 771–783, 2016.

[3] S. Henninger and V. Corrêa, “Software pattern communities: Current practices and challenges,” in *Proceedings of the 14th Conference on Pattern Languages of Programs*, ser. PLOP ’07. New York, NY, USA: ACM, 2007, pp. 14:1–14:19.

[4] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns:elements of reusable object-oriented software*. Addison-Wesley, 1995.

[5] F. Buschmann, R. Meunier, H. Rohnert, and P. Sommerlad, *Pattern-Oriented Software Architecture*. John Wiley and Sons, 1996.

[6] M. Fowler, *Patterns of Enterprise Application Architecture*. Addison-Wesley, 2002.