Advances in Computational Design, *Vol. 2, No. 4 (2017) 293-311*
DOI: https://doi.org/10.12989/acd.2017.2.4.293

Empirical assessment of design patterns' fault-proneness at different granularity levels

Mawal A. Mohammed[1] and Mahmoud O. Elish[*2]

[1]*Information and Computer Science Department, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia*
[2]*Computer Science Department, Gulf University for Science and Technology, Mishref, Kuwait*

(Received March 7, 2017, Revised September 14, 2017, Accepted October 1, 2017)

Abstract. There are several claimed benefits for the impact of design patterns (DPs) on software quality. However, the association between design patterns and fault-proneness has been a controversial issue. In this work, we evaluate the fault-proneness of design patterns at four levels: the design level, category level, pattern level, and role level. We used five subject systems in our empirical study. As a result, we found that, at the design level, the classes that participate in the design patterns are less fault-prone than the non-participant classes. At the category level, we found that the classes that participate in the behavioral and structural categories are less fault-prone than the non-participant classes. In addition, we found that the classes that participate in the structural design patterns are less fault-prone than the classes that participate in the other categories. At the pattern level, we found that only five patterns show significant associations with fault-proneness: builder, factory method, adapter, composite, and decorator. All of these patterns except for builder show that the classes that participate in each one of them are less fault-prone than the non-participant classes in that pattern. The classes that participate in the builder design pattern were more fault-prone than the non-participant classes and the classes that participate in several patterns: the adapter, the composite, and the decorator design patterns. At the role level, the most significant differences were between the classes that participate in some roles and the non-participant classes. Only three pairs of design pattern roles show significant differences. These roles are concrete-product vs. concrete-creator, adapter vs. adaptee, and adapter vs. client. The results recommend the use of design patterns because they are less fault-prone in general except for the builder design pattern, which should be applied with care and addressed with more test cases.

Keywords: design patterns; fault-proneness; software quality

1. Introduction

Design patterns are intended to encapsulate solutions to recurring design problems. They represent valuable design expertise that can be used in documenting and communicating these solutions. They were first introduced by Alexander *et al.* in their book Pattern Language in the context of architecture (Alexander *et al.* 1977). Later, the notion of design patterns was developed

∗Corresponding author, E-mail: elish.m@gust.edu.kw

Copyright © 2017 Techno-Press, Ltd.
http://www.techno-press.org/?journal=acd&subpage=7 ISSN: 2283-8477 (Print), 2466-0523 (Online)

maintenance: Comparing design patterns to simpler solutions", *IEEE Trans. Softw. Eng.*, **27**(12), 1134-1144.

Rudzki, J. (2005), "How design patterns affect application performance-a case of a multi-tier J2EE application", *Proceedings of the 4th international conference on Scientific Engineering of Distributed Java Applications*, Luxembourg-Kirchberg, Luxembourg.

Scanniello, G., Gravino, C., Risi, M., Tortora, G. and Dodero, G. (2015), "Documenting design-pattern instances: A family of experiments on source-code comprehensibility", *ACM Trans. Softw. Eng. Meth.*, **24**(3), 1-35.

SciTools (2014), http://www.scitools.com/download/.

Smith, J.M. and Stotts, D. (2003), *SPQR: Flexible Automated Design Pattern Extraction from Source Code*.

Tsantalis, N., Chatzigeorgiou, A., Stephanides, G. and Halkidis, S.T. (2006), "Design pattern detection using similarity scoring", *IEEE Trans. Softw. Eng.*, **32**(11), 896-909.

Vokac, M. (2004), "Defect frequency and design patterns: An empirical study of industrial code", *IEEE Trans. Softw. Eng.*, **30**(12), 904-917.

Vokac, M., Tichy, W., Sjoberg, D., Arisholm, E. and Aldrin, M. (2004), "A controlled experiment comparing the maintainability of programs designed with and without design patterns-a replication in a real programming environment", *Empir. Softw. Eng.*, **9**(3), 149-195.

Wohlin, C. (2013), "Empirical software engineering research with industry: Top 10 challenges", *Proceedings of the 1st International Workshop on Conducting Empirical Studies in Industry*.

TK