# Chapter 63
# An Empirical Study of the Effect of Design Patterns on Class Structural Quality

**Liguo Yu**
*Indiana University – South Bend, USA*

**Srini Ramaswamy**
*BU Power Generation at ABB, India*

## ABSTRACT

*Design patterns are standardized solutions to commonly encountered problems using the object-oriented programming paradigm. Applying design patterns can speed up software development processes through the reuse of tested, proven templates, or development paradigms. Accordingly, design patterns have been widely used in software industry to build modern software programs. However, as different design patterns are introduced to solve different design problems, they are not necessarily superior to alternative solutions, with respect to all aspects of software design. One major concern is that the inappropriate use of design patterns may unnecessarily increase program complexity, such as class structural quality. Theoretical analysis of the effect of design patterns on software complexity has been widely performed. However, little work is reported to empirically study how design patterns might affect class structural quality. This chapter studies six components from five open-source Java projects and empirically investigates if design patterns can affect class structural quality. The chapter finds that pattern-involved classes are more complex than pattern-free classes and recommends the cautious use of design patterns to avoid unnecessary increases in class complexity and decrease in class structural quality.*

## 1. INTRODUCTION

Design patterns are standardized experience-based solutions for commonly recurring object-oriented design problems (McNatt & Bieman, 2001; Shalloway & Trott, 2005). The use of design patterns can provide several advantages, such as increasing reusability, modularity, and quality of a program. Design patterns can also improve the consistency between program design and program implementation and

DOI: 10.4018/978-1-5225-3923-0.ch063

help achieve better coordination between the design and the implementation teams (Hasheminejad & Jalili, 2012). Since the introduction of design patterns (Gamma et al., 1995), software developers and researchers have identified over 20 patterns, many of which have been widely used in object-oriented programs. A recent study found that design patterns have also been widely used in open-source programs (Ampatzoglou et al., 2011a), which greatly facilitated the rapid growth and distribution of open-source projects.

Although design patterns can provide tested, proven solutions to representative and oft-occurring software design problems, they are not necessarily the best of class solutions with respect to software quality. First, while design patterns could be used to solve certain design problems, they might introduce new problems and reduce the system quality (Ampatzoglou et al., 2011a). Second, the use of design patterns should be context-based; an inappropriate use of design patterns could increase the system complexity and accordingly reduce software quality (Ampatzoglou et al., 2011b). To address this issue, extensive research has been performed in this area to study how design patterns can affect software quality, including reusability and maintainability (Ampatzoglou et al., 2011a; Ampatzoglou et al., 2011b). For example, theoretical analysis has been performed on several design patterns to see how they might affect complexity measurements (Huston, 2001). However, little work has been reported to empirically investigate the relationship between design patterns and software design quality.

This chapter presents a case study to investigate the relationship between class structural quality and design patterns. The study is performed on six open-source java components, which range from small to large applications. The class structural quality measurements of pattern-involved classes and pattern-free classes are calculated and compared.

The remainder of the chapter is organized as follows. Section 2 reviews related work in the quality analysis of design patterns. Section 3 presents the background knowledge of this study. Section 4 describes the data source and the data mining process. Section 5 presents the analysis and the results. Threats to validity are discussed in Section 6. Conclusions and future work appear in Section 7.

## 2. LITERATURE REVIEW

Because of the widespread use of design patterns in object-oriented software development, a large body of intensive research in this area has been published. For example, design patterns have been used to study the maintenance and evolution of software systems. Fushida et al. (2007) used pattern detection tools to study the change history of JUnit from the viewpoint of design pattern evolutions. Hsueh et al. (2011) proposed a method to evaluate how design patterns can effectively affect software evolution.

The work that is directly related to our study is to analyze the relationship between design patterns and software quality (Ampatzoglou & Chatzigeorgiou, 2007; Di Penta et al., 2008; Prechelt et al., 2001). For example, Zheng and Harper (2010) surveyed some concurrency design patterns and illustrated the relationship between concurrency design patterns and software quality attributes. Their study presented a mapping between concurrency design patterns and software performance and modifiability. Sandhu et al. (2008) proposed a set of quality metrics for various design patterns. Their study aimed to build high quality product through using high quality reusable design patterns.

Although extensive work has been reported, there are still no conclusive findings reported. Some studies showed that design patterns could reduce software complexity (Lange & Nakamura, 1995), and accordingly increase product quality. Some other studies found that design patterns could unintentionally