Information and Software Technology 000 (2017) 1–17

Contents lists available at ScienceDirect

Information and Software Technology

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

Are you smelling it? Investigating how similar developers detect code smells

Mário Hozano[a,b,*], Alessandro Garcia[c], Baldoino Fonseca[d], Evandro Costa[d]

[a] Department of Computing Systems, UFCG, Campina Grande-PB, Brazil
[b] Núcleo de Ciências Exatas, UFAL, Arapiraca-AL, Brazil
[c] Opus Research Group – LES, Informatics Department, PUC-Rio, Rio de Janeiro-RJ, Brazil
[d] Computing Institute, UFAL, Maceió-AL, Brazil

ARTICLE INFO

Article history:
Received 8 December 2016
Revised 29 August 2017
Accepted 5 September 2017
Available online xxx

Keywords:
Code smell
Detection
Software maintenance
Empirical study

ABSTRACT

Context: A code smell indicates a poor implementation choice that often worsens software quality. Thus, code smell detection is an elementary technique to identify refactoring opportunities in software systems. Unfortunately, there is limited knowledge on how similar two or more developers detect smells in code. In particular, few studies have investigated if developers agree or disagree when recognizing a smell and which factors can influence on such (dis)agreement.
Objective: We perform a broader study to investigate how similar the developers detect code smells. We also analyze whether certain factors related to the developers' profiles concerning background and experience may influence such (dis)agreement. Moreover, we analyze if the heuristics adopted by developers on detecting code smells may influence on their (dis)agreement.
Method: We conducted an empirical study with 75 developers who evaluated instances of 15 different code smell types. For each smell type, we analyzed the agreement among the developers and we assessed the influence of 6 different factors on the developers' evaluations. Altogether more than 2700 evaluations were collected, resulting in substantial quantitative and qualitative analyses.
Results: The results indicate that the developers presented a low agreement on detecting all 15 smell types analyzed in our study. The results also suggest that factors related to background and experience did not have a consistent influence on the agreement among the developers. On the other hand, the results show that the agreement was consistently influenced by specific heuristics employed by developers.
Conclusions: Our findings reveal that the developers detect code smells in significantly different ways. As a consequence, these findings introduce some questions concerning the results of previous studies that did not consider the different perceptions of developers on detecting code smells. Moreover, our findings shed light towards improving state-of-the-art techniques for accurate, customized detection of code smells.

1. Introduction

A code smell indicates a poor implementation choice that often worsens software quality [1]. Therefore, code smell detection is an elementary technique for supporting a wide range of quality improvement tasks, such as enhancing program comprehension [2], reducing program error proneness [3], and combating software design degradation [4]. However, detecting code smells in practice is much harder than it is usually assumed or advertised [5–8]. A key, prevalent challenge for developers is to perform this task on source code produced by others.

Recent estimates confirm at least 180 billion lines of legacy, smelly code are target of software refactoring [9]. In such systems, either experienced or inexperienced developers have to reason about the smelliness of the legacy code. In the context of open-source projects, the presence of code smells is a frequent reason on why pull requests are rejected by core developers [10]. Different core developers have to judge the "smelliness" of the source code produced by potential contributors in a single project [10]. As core and peripheral developers have varying experience and backgrounds, their views about each code smell may conflict with each other. Moreover, it is a commonplace for developers to be shuffled between projects, thereby requiring them to detect smells in unfamiliar code.

* Corresponding author at: Department of Computing Systems, UFCG, Brazil.
E-mail address: hozano@gmail.com (M. Hozano).

http://dx.doi.org/10.1016/j.infsof.2017.09.002
0950-5849/

All these scenarios expose developers to formulate their (possibly diverging) views about code smells in a single project. In particular, the informal and subjective definition of certain smell types [1] may lead two or more developers to reason about each smell occurrence differently [11]. In spite of the extensive tool support for smell detection available nowadays (e.g. [12–18]), developers still need to analyze each smell individually and confirm its occurrence on the system. While a developer may confirm a code snippet as the host of a particular smell, other developers may not necessarily agree. For instance, consider the *Long Method* smell defined as *a method that is too long and tries to do too much* [1]. When a developer is focused on detecting *Long Methods*, he may face subjective questions, such as:

- How to judge whether a method is long?
- How to judge whether a method is doing too much?
- Is it possible to detect a *Long Method* solely based on the lines of code of a method?
- How many lines of code are required to characterize a method as *Long Method*?

Given the subjective nature of such questions, different developers working on the same code base may have different answers to them. As a consequence, they may or may not agree on the occurrence of a *Long Method*. Similar reasoning applies to the detection of other smell types. Many would claim high agreement is beneficial for various reasons. First, it would be easier to promote consistency across code reviews. Second, it would be more feasible to rely on existing tooling support for smell detection (e.g. [12–18]). On the other hand, these benefits become harder to achieve if high disagreement across developers is the norm, rather than the exception.

In either case, little is known about how similar developers detect smells in code. In particular, there is no understanding if certain factors influence or not (dis)similar opinions of developers on smell occurrences. Various factors underpinning developers' characteristics may play a role on the code smell detection process. Such factors may include basic characteristics of software developers, such as similar (or diverse) background and experience. Independently from a developer's profile, a more personal factor may also influence how developers detect code smells. For instance, each developer can rather follow their own very specific way (from herein called "heuristic") to detect a code smell.

However, the factors influencing such (dis)agreement of developers have been rarely studied. Only few studies [19–22] have only investigated the level of (dis)agreement among developers on smell occurrences. The studies described in [19,21,22] have analyzed the agreement among developers, but their findings were limited to only three types of smells. One might expect developers detect more similarly certain types of code smells than others. For instance, there are several types of code smells affecting different structures of a program, ranging from a simple statement within a method to a group of classes [1]. Developers might tend to have a more uniform view on smells confined to simpler program structures. The study presented in Mäntylä et al. [20] investigated the developers' agreement over a greater variety of code smells. The authors performed a preliminary analysis if developers' background and experience could influence their agreement. However, the small data set used in this study did not allow the authors to obtain conclusive results. More importantly, none of these studies have investigated how similar are the heuristics formulated by different developers to detect smells.

In this context, this paper reports a broader study aiming at investigating the (dis)agreement among developers on detecting occurrences of 15 different smell types. The study also analyzes whether certain factors may influence such (dis)agreement. We count with 75 developers who evaluated the presence of code smells into a huge set of code snippets from real projects. Altogether, more than 2700 evaluations were collected and analyzed. We assessed the agreement among developers by considering its statistical significance. We analyzed the influence of six factors on the developers' agreement. Such factors are mainly related to developers' background and experience. In addition, we investigated how similar are the heuristics formulated by different developers to detect smells.

Our study led to three main findings: (i) developers presented statistically significant low agreement levels on evaluating all 15 smell types, contradicting key results from previous studies; (ii) the developers' background and experience did not present a consistent influence on the agreement; and (iii) the heuristic factor played the most important role on developers' agreement. These findings suggest the increasing need for improving customizable techniques for smell detection by taking into account the perception of each developer. Thus, we discuss the potential and possible limitations of state-of-the-art techniques for smell detection.

The remaining of this document is structured as follows. Section 2 describes the design of our empirical study and the research questions. In Section 3 we present the results of the study and, in Section 4, we answer the research questions. Section 5 details the threats of the study. Next, Section 6 presents the related work. Finally, Section 7 presents the conclusions observed in our study.

2. Study design

This study aims at investigating how similar the developers detect code smells in unfamiliar source code. We analyze if developers tend to agree or disagree on the occurrences of smells pertaining to a wide range of types. In particular, we study to what extent certain factors may or may not influence common perceptions shared by different developers. Developers detected several instances of 15 different types of code smells found into real systems. As a result, we analyzed the agreement among developers according to their evaluations. Moreover, we investigated if participants with common characteristics detected code smells similarly. In this way, two main research questions guided our study:

- **RQ1**: *Do developers agree on the "smelliness" of the source code?*
  The motivation for this question is to investigate if, in general, developers agree about detecting code smells into real projects. In addition, we analyze the degree of such agreement in order to verify how differently the developers detect smells in the same programs. Particularly, such analysis becomes difficult because it requires the participation of several developers with different characteristics (Section 1) in order to create a relevant sample. Such requirements may have reduced the conclusions of previous studies that performed investigations concerning similar questions [19,20]. Thus, we aimed at performing a deeper investigation in order to increase the knowledge about how similar the developers detect smells in unfamiliar source code. Our results may shed light on the construction of more efficient detection techniques.
- **RQ2**: *What makes developers agree on the "smelliness" of the source code?*
  The motivation of this question aims at analyzing if certain factors may influence (dis)agreement of developers on smell detection. In particular, we analyze whether developers detect code smells similarly, when grouped according to common characteristics that they share. Previous studies have performed similar investigations grouping developers according to their experience and background [19–22]. In addition to these aspects, we also investigate how the (dis)similar judgment of developers is influenced by heuristics they formulate for smell

Please cite this article as: M. Hozano et al., Are you smelling it? Investigating how similar developers detect code smells, Information and Software Technology (2017), http://dx.doi.org/10.1016/j.infsof.2017.09.002