Detecting Java Programming Language Smell Code with a Long Parameter List

Fajar Ratnawati and Jaroji
Politeknik Negeri Bengkalis, Bengkalis, Riau, Indonesia

Keywords: Smell Code, Java, Long Parameter List.

Abstract: This research resulted in the evaluation of the smell code (wrong program code) in the Java programming language. The main objective to be achieved in this study is to detect the smell code (wrong program code) so that the quality level of the code can be seen. The smell code that will be detected in this study is a long parameter list. To support the research process, the evaluation phase of a Java-based application case study was taken from the PMD static code analyzer as a code inspection sample for smell code detection and software metric calculations. The results obtained are code detection based on smell code rulesets, namely warning errors that have been determined with a number of parameters greater than four.

1 INTRODUCTION

The complexity of software systems is currently increasing at such a rapid pace that software companies must constantly update the source code that has been created (D. Di Nucci et al, 2018). These ongoing changes frequently take place under duress and in a short period of time. As a result, developers abandon sound programming practices and principles. They want to provide the best product possible but are unable to do so because they are working in the shortest amount of time possible (F. Shull et al, 2013). In other words, the source code's quality is not optimal or good because of an error or problem in the source code.

Java is a popular programming language because the range of applications that can be made using this language is very wide, from computers to smartphones (Mardison, 2017). The Java programming language is known as OOP-Object Oriented Programming. Java was first developed by Sun Microsystems, which was started by James Gosling and released in 1995. Currently, Sun Microsystems has been acquired by Oracle Corporation. Java is Write Once, Run Anywhere (a program that is written once and can run on multiple platforms). Just like programming in general, the Java programming language can work using a database.

Almost all of the Study Programs in the Informatics Engineering Department use the Java programming language for the practicum learning process. Hundreds of Java program codes, the result of student assignments, are still manually checked. Sometimes, only checked by the output generated from the code without regard to the quality of the program.

Smell code is a term used to indicate that there may be a problem in a program's code. (Slinger and Stevan, 2005). A smell in code indicates that something is wrong, or that it should not be in the code. Smell code is a problem caused by a design flaw in a piece of software. This poor design can lower the quality of the compiled software. Smell code must be reduced by detecting its presence in program code and repairing it as needed. Humans can detect smell codes manually, or a system can do it automatically. However, manual detection will take a long time, especially if hundreds of program codes are to be detected. As a result, we require a code smell detector device that can be used automatically to detect hundreds of program codes in a relatively short period of time.

The long parameter list is one of the methods in the smell code. There is no specific rule on how many parameters to use in one method. Usually, more than three or four is considered too many. Long parameters that are easy to recognize include:

1. It is hard to use a method call or to get the parameters in the correct order.
2. It is hard to read and interpret what a method call does.
3. A method call has boolean parameters.

Ratnawati, F. and Jaroji, .
Detecting Java Programming Language Smell Code with a Long Parameter List.
DOI: 10.5220/0011755100003575
In *Proceedings of the 5th International Conference on Applied Science and Technology on Engineering Science (iCAST-ES 2022)*, pages 228-231
ISBN: 978-989-758-619-4; ISSN: 2975-8246

4. A method call has null parameters as optional parameters.

Long parameter lists can occur due to trying to do too many things in a method or trying to minimize dependencies between objects.

To realize the ideas outlined in the background above, there are problems that will be researched and found the right solution model. These problems are as follows:

1. How to design in building a system for detecting smell code.
2. How to implement a smell code detection system using long parameter list.
3. How to detect error code or bad code (smell code) in the Java programming language.

This study focuses on detecting smell codes with the type of bad smell long parameter list for the Java programming language.

2 LITERATURE REVIEW

There have been several studies related to the topic of discussion so far, which later became development material. Sujadi's research seeks to identify smell codes and anti-patterns in case studies of E-Commerce applications that manage goods and customer master data, as well as sales and payment transactions (Sujadi, 2019).

Putro and Liem's research aims to detect code smells that can be used for all programming languages, particularly to evaluate the outcomes of practical work on programming lectures, using data in xml format and a by-rules approach (Putro and Liem, 2010). The research of Firdaus et al aims to detect a refused bequest type of smell code during the design stage (Firdaus et al., 2018).

J. Kreimer's research uses Decision Trees as an approach to classifying and building a knowledge base. The training and detection phases are carried out independently. The knowledge base model is built in the training phase. The model represents the result of accumulated knowledge that was previously learned based on input from users. The study also uses software metrics as a feature to identify bad smells. Some examples of bad smells that are targeted to be detected in this research are long method and big class. (Kreimer, 2005).

3 PROPOSED FRAMEWORK SMELL CODE

The proposed smell code framework consists of four stages, namely collecting java projects, creating data sets, creating java code detection rules, and smell code detection systems. The smell code framework can be seen in the following figure.

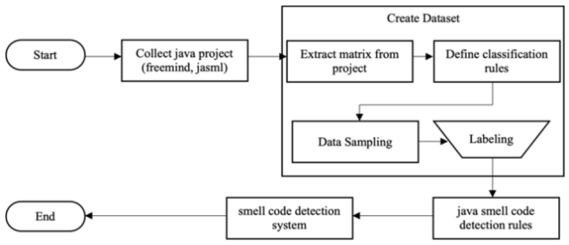

Figure 1: Proposed Framework Smell Code.

Collecting java projects as data sources, this project was obtained from Qualitas Corpus which is a collection of heterogeneous or unique software systems used in empirical studies related to Java code artifacts. In this study, two projects were used as data sources, namely freemind and jasml.

Next, create a dataset from the downloaded java project (freemind & jasml) using the WekaNose tools, there are several stages in making this dataset.

a. Extraction of metrics or calculate each metric in the java project, which will be used as a feature in machine learning.
b. Determine the rules that will categorize the dataset into normal (false), and smell (true).
c. Manually labelling true/false on the generated dataset.
d. Build a detection rule model

The results are used as rules on the smell code detection system. For more details can be seen in the following image.

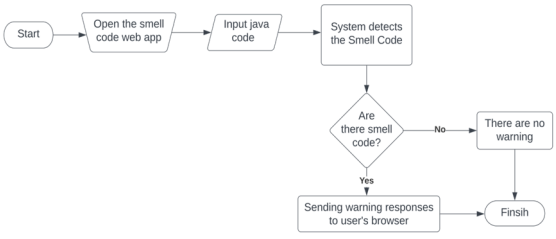

Figure 2: System Flow Built.

The picture above describes the workflow of the system created, namely:

1. The user opens the web page of the smell code app detector.

2. The user enters a line of Java code and presses the analyze button.
3. The line of code will be sent to the server.
4. The server will analyze the line of code based on the pre-defined rules.
5. If a smell code is found in the code, a warning message will be sent along with the smell code details to the user's browser.
6. Done

4 DESIGN RESULT

The result achieved in this research is a smell code detection system for the Java programming language. This system is web-based and has a user-friendly interface.

Figure 3: Initial View.

The picture above is the initial appearance of the smell code system. There is a window to enter the Java code and an analyze button to check whether there is a smell code or not. After the Java code is entered, the next step is to press the analyze button.

Figure 4: Java Code Example.

The figure depicts an example of the program code that is entered, which contains a long parameter list in the constractor method. As a result, the system able to detect the smell code. Figure 4 indicates the result of detecting the smell code.

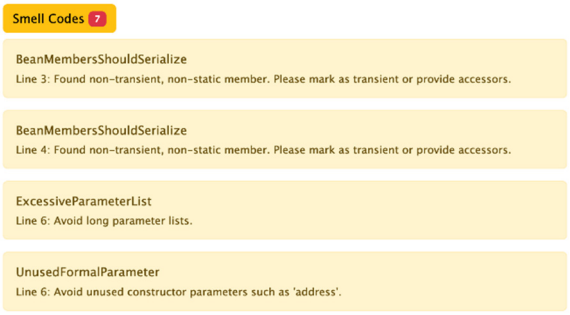

Figure 5: The Resulting Output.

There were seven smell codes detected in the Java code, including members serialize problem, long parameter list, and variable unused. User can fix their java code and try to check it again using this system.

If the Java code detects a long parameter list, then the things you can do are as follows:

1. If the parameter can be obtained from another object, replace the parameter with a method call. The object can be a field in the class or passed as a parameter.
2. If the parameters belong to a single object, preserve the whole object.
3. If the parameters come from different objects, introduce a parameter object.
4. If there is a boolean parameter, consider replacing the parameter with explicit methods.

The benefit of long-side parameter list detection is that it improves code readability and reduces duplication. A long parameter list is sometimes needed for some programmers for some reason. So, smell code detection also does not guarantee that all Java code will be free from problems.

5 CONCLUSIONS

This system makes it easier for users to check the smell code of Java code so that it can assist in building applications in the Java programming language. Using a long parameter list can reduce the number of parameters in a method and avoid duplicates.

REFERENCES

D. Di Nucci, F. Palomba, D. A. Tamburri, A. Serebrenik and A. De Lucia, "Detecting code smells using machine learning techniques: Are we there yet?," 2018 IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER), 2018, pp. 612-621, doi: 10.1109/SANER.2018.8330266.

F. Shull, D. Falessi, C. Seaman, M. Diep, and L. Layman, Perspectives on the Future of Software Engineering. Springer, 2013, ch. Technical Debt: Showing the Way for Better Transfer of Empirical Results, pp. 179–190.

Firdaus, M.F., Priyambadha, B., Pradana, F. (2018). Pembangunan Sistem Untuk Pendeteksian Code Smells Refused Bequest. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer. Vol. 2, No. 12

J. Kreimer, "Adaptive Detection of Design Flaws," Electronic Notes in Theoretical Computer Science, pp. 117-146, 2005.

M. Fowler, K. Beck, J. Brant, W. Opdyke dan d. Roberts, Refactoring: Improving the Design of Existing Code, Boston: Addison-Wesley Longman Publishing Co., Inc, 1999.

Mardison, 2017. Perancangan Sistem Penunjang Keputusan Untuk Mengoptimalkan Pemberian Kredit Pada Bank Bpr Kubang Dengan Bahasa Pemrograman Java Dan Didukung Dengan Database Mysql. Jurnal Processor. Jilid 7. Terbitan 1

Putro, H.P., Liem, I., (2010). Deteksi Code Smell Pada Kode Program Dalam Representasi Ast Dengan Pendekatan By Rules. Seminar Nasional Aplikasi Teknologi Informasi 2010 (SNATI 2010). Yogyakarta

Slinger, Stevan, 2005. Code Smell Detection in Eclipse. Delft University of Technology, Delft, Belanda.

Sujadi, S.F. (2019). Evaluasi Deteksi Smell Codedan Anti Patternpada Aplikasi Berbasis Java. Jurnal Teknik Informatika dan Sistem Informasi. Volume 5 Nomor3.