THE ANALYSIS OF SOFTWARE SOURCE CODE READABILITY: CASE STUDY AT EDUCATION OF INFORMATICS AND COMPUTER ENGINEERING STUDY PROGRAM OF SEBELAS MARET UNIVERSITY

Imam Setiawan[1], Dwi Maryono[2], Basori[3]
[1,2,3] Department of Informatics Education, Sebelas Maret University

Article Info

Article history:

Received Jan 20, 2019
Revised Feb 10, 2019
Accepted Feb 22, 2019

Corresponding Author:

Imam Setiawan,
Department of Informatics Education,
Sebelas Maret University,
Jl Ahmad Yani, no 200,
Pabelan, Kartasura, Surakarta,
Jawa Tengah, 57169, Indonesia.
Email: s.imam29@gmail.com

ABSTRACT

Source code readability is a property that influences how easily a given piece of code can be read and understood. Since source code readability can affect software quality, especially maintainability, then programmers must have a good sense of writing readable code. For computer science and software engineering student, they have to start learning how to write readable code in order to compete later in the industrial. Unfortunately, computer science and software engineering curriculum promote understanding the programming paradigms of a particular language, compared to write readable code. Based on its importance, we analyzed the source code of software written by Education of Informatics and Computer Engineering of Sebelas Maret University students to describe its readability. We determine readability category from source code based on two programming features, variable and function writing. Each programming features involved has its own criteria so that it can be classified in the readable or less readable category. Finally, we discuss the implications of this study on the learning process of Education of Informatics and Computer Engineering study program, Sebelas Maret University. For example, our data suggest using code reviews to teach the student about source code readability.

Keywords: source code readability, identifier naming, function writing

DOI: https://doi.org/10.20961/joive.v2i1.35695

1. INTRODUCTION

Readability of the source code can be defined as a human judgment about how easily someone understands programming code (Buse & Weimer, 2009). Source code writing skills is important expertise in competing in the software industry. By teaching source code writing skills that is easy to read will give more value to students of Information and Computer Engineering majors in competing in the industrial world later.

But in most programming learning in college, students are only taught to write source code that can run regardless of its readability. This is because Computer Science curricula promote to have an understanding of programming paradigms of programming languages (e.g. functional and non-functional), compared to write readable code (Sedano, 2016).

Ability to write source code that is easy to be read even though needed in complete various student assignments in lecture. Ease of reading source code software written by college student will help the performance lecturer in evaluating software development task results from the student. In the evaluation process, the lecturer review software source code then verified to college student to detect its fraud in doing the assignment. This is a manner directly will facilitate the work process from the lecturer.

This research specifically focuses on the readability of the source code software project written by Sebelas Maret University (UNS) Education of Informatics and Computer Engineering (PTIK) students in order to explain the readability condition of the source code as well as ways to improve it. Source code of software project is taken from the results of software projects that have implemented the Object-Oriented Programming paradigm (OOP) both from the final assignments of courses, industrial practices, and final assignments. The programming features that are analyzed are variables and functions because they are the basic features that affect other features.

2. RESEARCH METHOD

The study took place in September 2018 until May 2019 located in the Education of Informatics and Computer Engineering study program, Sebelas Maret University, having its address at Ahmad Yani Street Ahmad Yani No 200A, Pabelan, Sukoharjo Regency. The research methodology is descriptive qualitative through document analysis techniques and interviews with PTIK UNS students. The documents analyzed were in the form of software projects resulting from student work both from the assignments of subjects, industrial practice and final assignments.

The data analysis technique uses an interactive analysis model consisting of data collection, data reduction, data presentation and conclusion drawing. The interactive analysis model scheme can be seen in Figure 1.

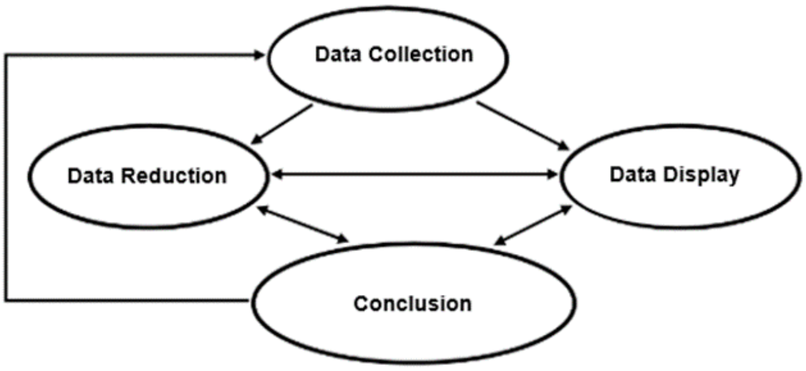

Figure 1. Interactive Analysis Model (Sutopo, 2006)

Data collection is done by copying software projects from students of Education of Informatics and Computer Engineering study program, Sebelas Maret University then to be interviewed later on the results of analysis of related projects. The process of data reduction is done by measuring the readability of each project using several software readability criteria. In addition, data reduction is also done by taking the essence of interviews with student project sample software owners.

The variable naming criteria used as a reference in the analysis include the use of the word whether it matches the variable value, the number of words that make up the word, and the naming style used. While the criteria used as a reference in the function analysis include the suitability of the word used in the name of the function, writing conditional structure and repetition, and writing statements whether it has applied the concept of Do One Thing.

The results of the analysis are then grouped and presented in the form of a percentage of each programming feature. After that, conclusions are drawn based on the data presented to find answers to the readability conditions of the source code written by students of Education of Informatics and Computer Engineering study program, Sebelas Maret University.

3. RESULT AND ANALYSIS

In the research in the UNIK PTIK study program, data was obtained that students of Education of Informatics and Computer Engineering study program, Sebelas Maret University were still practising writing less readable source code.

3.1. RESULT

In terms of variable readability, as many as 140 of the 392 different variable names or around 35% are included in variables with low readability. Criteria for causes of low readability of these variables are grouped into two categories, non-informative and disinformative.

Non - informative means that a variable name does not contain instructions or information in explaining the contents of the variable. From research findings, variables that are classified as non-informative using serial words, unusual abbreviations and single letters as variable names.

Disinformative means that a variable name contains information but still does not explain clearly or even disguise the contents of the variable. From the research findings, disinformative variables use names from abstract words and words that have too general/broad coverage without being followed by more specific words.

Table 1. Percentage of low readability variable criteria

Category	Criteria	Quantity	Percentage	Example
Non-informative	Serial words	23	16%	data2
	Unusual abbreviations	36	26%	res (result)
	Single letters	3	2%	c
Disinformative	Generic words	41	29%	data
	Abstract words	37	27%	where
TOTAL		140	100%	

3.2. ANALYSIS

In terms of function readability, as many as 168 from 335 functions or around 50.1% is included in the function with low readability. Forty-eight (48) function caused by less descriptive naming function, 84 function for writing statements on a function that does not meet the concept of Do One Thing and the remaining 101 due to the low readability variables in it.

Based on research findings, there are 3 criteria for the naming function which includes a less descriptive naming category. First, the use of words to show certain concepts that are not consistent. For example, it is found that two functions that are on a project have the same operation but are named with different words ("insert" and "add_user"). Second, use words/phrases other than verbs/verb phrases. Third, the use of English words that are not in accordance with grammar so that it creates ambiguous meanings if interpreted.

While in terms of writing statements function, a function written in more than one level of abstraction will be considered a function that works more than one thing so that it is categorized as not fulfilling the concept of Do One Thing.

Table 2. Percentage of low function readability criteria

Criteria	Quantity	Percentage
Non-descriptive name	48	29%
Not applied *Do One Thing* concept yet	84	50%
Low variable readability	101	60%

Apart from the readability factor variable within a function, in general still found function with lower readability as well as naming a less descriptive and does not fit the concept of Do One Thing. Whereas in terms of conditional structure and repetition, the writing has followed the recommended criteria.

4. CONCLUSION

Source code readability can be measured from various programming features. In this study, the researcher focused on feature variables and functions to determine the readability of the source code written by students of Education of Informatics and Computer Engineering study program, Sebelas Maret University. From the results of the study, it can be concluded that the readability of the source the code of the software project written by students of Education of Informatics and Computer Engineering study program, Sebelas Maret University still needs to be improved including in terms of naming variables, naming functions, simplifying functions and writing comments.

In addition, a way to improve the readability of the source code of a software project written by students of Education of Informatics and Computer Engineering study program, Sebelas Maret University can be done by introducing writing standards source code, give code example with good readability and applying learning methods such as code review.

Based on the research conclusions, it is necessary to insert material about the readability of the source code in lectures, starting from basic features such as variables, writing functions, layout settings (formatting), management of conditional and logic structures, comment writing and so on. Lecturers are expected to have mastered the basic techniques of the source writing good code in order to teach source readability material code and evaluate the readability of the source code college student. While students are advised to begin to foster awareness of the importance of readability of the source code and learn writing techniques that can improve the readability of the source code.

REFERENCES

[1] Buse, R. P., & Weimer, W. R. (2009). Learning a metric for code readability. *IEEE Transactions on Software Engineering, 36*(4), 546–558.

[2] Gruber, H., & Jegatheeswaran, V. (2018). *The Improvement of Understanding and Readability of Code Through the Application of Programming Guidelines*.

[3] Sedano, T. (2016). Code Readability Testing, an Empirical Study. *IEEE 29th International Conference on Software Engineering Education and Training (CSEET)*, 111–117.

[4] Sutopo, H. B. (2006). *Penelitian kualitatif: Dasar teori dan terapannya dalam penelitian*. Surakarta: Universitas Sebelas Maret.