Table of Contents

[Effect of class size on Software Maintainability 3](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667805)

[Section 1 3](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667806)

[Introduction 3](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667807)

[Section 2 3](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667808)

[GQM Approach 3](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667809)

[Section 3 4](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667810)

[Subject Programs 4](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667811)

[Gathered Datasets 5](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667812)

[Section 4 6](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667813)

[Tool Used 6](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667814)

[Data Collection Tool 6](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667815)

[CK Metrics Tool 6](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667816)

[Section 5 7](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667817)

[Results and Findings 7](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667818)

[Section 6 14](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667819)

[Conclusion 14](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667820)

[References 14](file:///C:\Users\nishm\Downloads\Ms%20Nishma%20Lewis%2023%20-%20OOD%20-%20Week%202%20-%20Group%20-%20Assignment1%20(1).docx#_Toc131667821)

# Effect of class size on Software Maintainability

# Section 1

## Introduction

One of the most important features is maintainability in software systems. Software maintainability refers to the ability of a software system to be modified, corrected, or adapted to meet changing requirements or environments. In software development, the maintenance phase consumes a significant portion of the overall cost and effort involved. So it is a critical aspect of software engineering. Class size is the internal class quality attribute and researchers utilize different measuring ways to measure the class size [1]. Larger classes tend to be more complex and harder to understand, which can make it difficult to modify or extend the class in the future. When modifying the larger class then it introduces bugs. The effect of class size on software maintainability is an important consideration for software developers, as it can have a significant impact on the ease and cost of maintaining a software system.

Chidamber and Kemerer (1994) proposed the metrics such as WMC, DIT, LOC, CBO, LCOM, and RFC for measuring class size. One of the CK metrics, called Weighted Methods per Class (WMC) is closely related to class size [1]. It measures the complexity of a class by counting the number of methods defined in the class and weighting them by their complexity. Other than WMC, LCOM (Lack of Cohesion in Methods) measures the number of disconnected sets of methods in a class. If the class is too high then LCOM becomes high which indicates a low level of cohesion and the class should be split into smaller. LCOM is considered for ensuring maintainability. For the current empirical research work, both WMC and LCOM values are taken into the account.

# Section 2

## GQM Approach

**Scope and Objectives**

The objective of this report is to investigate the effect of class size on software maintainability. It aims to measure the relationship between class size (internal quality attributes such as size, coupling, and cohesion) and software maintainability (external quality attributes) and how this relationship impacts the software development process. The proposed work analyses how different class sizes affect the maintainability of software and identify the best practices for managing class size to ensure optimal software maintainability. The relationship between class size and maintainability in object-oriented programming languages such as Java is measured by considering CK metrics values. It also provides recommendations for developers and software organizations on how to manage class size effectively and efficiently to improve software maintainability.

**Research Questions**

1. What is the effect of class size on software maintainability?
2. How does the relationship between class size and software maintainability impact the development process?
3. What are the best practices for managing class size to ensure optimal software maintainability through CK metrics?

**Metrics**

Class Size – the number of lines of code in a class. It is essential to limit the class size in the code. WMC (Weighted Methods per Class) is used to measure the class complexity. If the WMC value is higher, the class is too complex so it is difficult to maintain the code. Limit the class size by extracting methods.

Maintainability - A measure of how easy it is to maintain software. Maintainability Index estimation calculation is focused on by researchers for estimating the maintainability metrics [3]. It is essential to measure the impact of class size on the software development process.

LCOM (Lack of Cohesion in Methods) is used to measure class cohesion [4]. A high LCOM indicates a lack of cohesion.

# Section 3

## Subject Programs

A dataset is a collection of structured data that is organized in a particular way. Datasets are used in different processes for completing empirical research work. The gathered dataset has made up of one or more features that describe the characteristics of that instance. For the proposed empirical research work, five datasets are gathered by utilizing Github. The research work focuses on maintainability issues due to a lack of CK metrics and its standard value.

**Criteria Selection**

For selecting the dataset, three points or criteria are fixed that is,

* The size of the selected dataset is above 10k. It is essential to consider the larger dataset for empirical research work because it helps to analyze and produce results accurately.
* Developers involved – A minimum of 5 developers should be included in the development. The gathered datasets can be handled by many contributors.
* Year of code – The developed code is considered to be at least 3 years old. The selected dataset is updated currently. The right code is focused on dataset selection because the current empirical research work is considered maintainability so there is a possibility of updates regularly.

## Gathered Datasets

Table 1- Gathered Datasets and their features

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Parameters** | **Arthas** | **Libsu- Master** | **Miaosha –Master** | **Nacos-Develop** | **React-Native** |
| Link | <https://github.com/alibaba/arthas> | <https://github.com/topjohnwu/libsu> | <https://github.com/qiurunze123/miaosha> | <https://github.com/alibaba/nacos> | <https://github.com/henninghall/react-native-date-picker> |
| Programming language | Java | Java | Java | Java | Java |
| Program Description | It is a Java diagnostic tool and it is an open-source tool by Alibaba. It allows developers to troubleshoot issues without modifying code for Java applications. | It uses root permissions and provides a complete solution for applications. | Internet engineer advanced analysis is focused. | For building cloud-native applications it is an easy-to-use dynamic service discovery, configuration, and service management platform. | It is a video component. |
| Size | 28983 | 23643 | 65728 | 45429 | 11367 |
| Year | 2018 | 2018 | 2018 | 2018 | 2018 |
| Contributors (Developers) | 170 | 8 | 9 | 304 | 239 |

# Section 4

## Tool Used

## Data Collection Tool

***GitHub***

GitHub is open-source software and it has a collection of open-source software projects relevant to the research topic [5]. It is a useful tool for gathering the requested datasets. The GitHub repositories help to gather relevant datasets and GitHub allows the developer to actively maintain a code for a long time. GitHub ensures ethical considerations when gathering datasets for completing the research work.

## CK Metrics Tool

**Mauricioniche**

Mauricio Aniche is a professor and researcher at the University of São Paulo in Brazil, and he has developed a tool called "ckjm" which stands for "Chidamber and Kemerer Java Metrics". This tool supports for implementation of the CK metrics suite specifically designed for calculating software metrics for Java programs. “ckjm” helps to calculate six metrics for java classes that are WMC (Weighted Methods per Class), DIT (Depth of Inheritance Tree), NOC (Number of Children), RFC (Response For a Class), CBO (Coupling Between Object Classes), and LCOM (Lack of Cohesion in Methods) [6]. One of the advantages of ckjm is that it can be easily integrated into automated build and test systems. It is available as an open-source project on GitHub, and it can be used either as a command-line tool or as a library that can be integrated into other software tools like Visual Studio Code. CKJM in visual studio helps to extract the metrics automatically. It helps to change the metric values by calculating the metrics.

***Visual Studio Code***

Visual Studio Code is a free and open-source code editor developed by Microsoft. It is a highly popular tool for software development due to its simplicity, versatility, and powerful features [6]. The proposed work utilizes Visual Studio Code which is used to extract CK metrics for a software project. Visual Studio code has several benefits such as version control, allowing for writing and editing code, debugging, integrated terminal, and extensions. The proposed research work focuses on maintainability metrics by extracting metrics.

# Section 5

## Results and Findings

The relationship between class size and maintainability is determined by generating tables and graphs for gathered datasets.

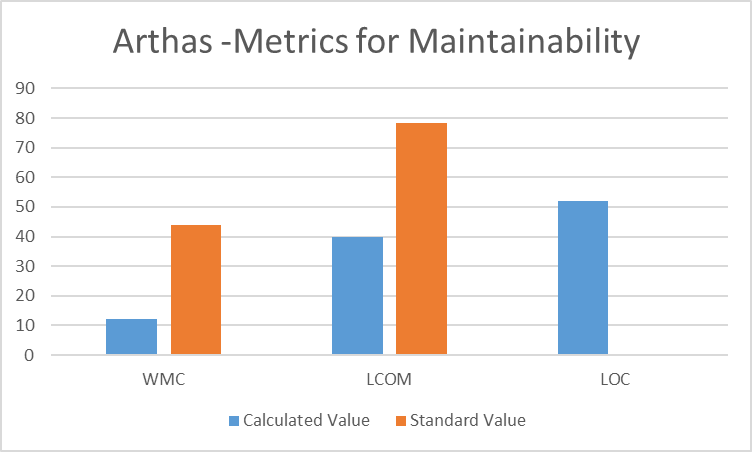
WMC helps to measure the complexity of the individual class. The sum of the complexity of the methods is defined through WMC. This value helps to predict how much time and effort is required for developing the code and maintaining the code [7]. WMC greatly affects the class size of the code. The developers should consider the standard values when developing classes because it affects the class size. The standard value for WMC is less than 49.

LOC (Lines of Code) is used to measure the software size. The number of lines included in a program is counted for calculating the LOC. There is no standard value for the lines of code. The physical lines of code of the source code should be measured for calculating the LOC.

There is a negative correlation between class size and the CK metric of Lack of Cohesion in Methods (LCOM). LCOM measures the lack of cohesion among the methods of a class. If the LCOM value is higher, it lower software maintainability. A class with high LCOM may be more difficult to maintain because it introduces bugs or errors. The cost of maintenance and timing will be increased because they need to spend more time testing the code or maintaining the code. The standard value for LCOM is less than or equal to 78.34.

Table 2- Arthas -Metrics for Maintainability

|  |  |  |  |
| --- | --- | --- | --- |
|  | **WMC** | **LCOM** | **LOC** |
| **Calculated mean Value** | 12.19716 | 39.85954 | 51.97036 |
| **Standard Value** | < 49 | <= 78.34 |  |

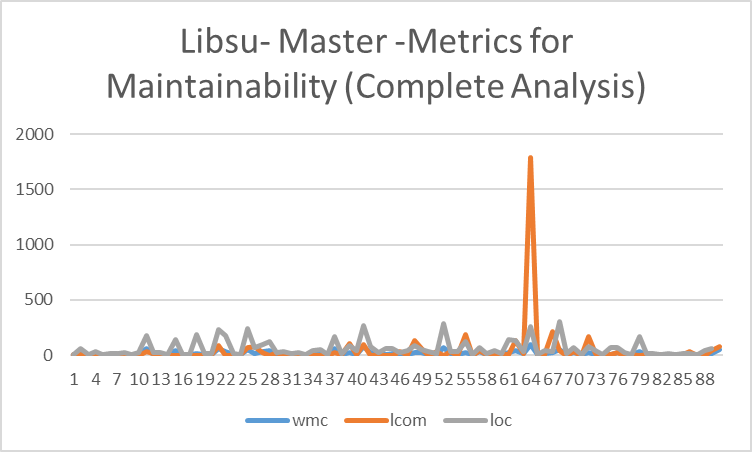


Graph 1-Arthas -Metrics for Maintainability

The “Arthas” project code is selected and metrics for the code are generated by utilizing Visual Studio Code. For the empirical research work WMC, LCOM, and LOC are taken into account. The graph is generated after calculating the average value for these metrics. The graph is plotted against the standard value and calculated average value. When compared to the standard value, the calculated average value is less. But the individual class value for the code is high so the developers need to manage these metrics values to ensure higher maintainability.

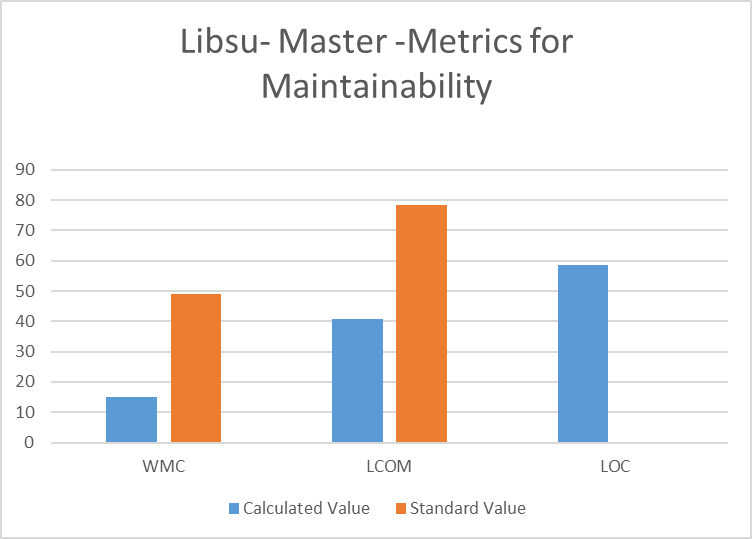
Table 3-Libsu- Master -Metrics for Maintainability

|  |  |  |  |
| --- | --- | --- | --- |
|  | WMC | **LCOM** | **LOC** |
| **Calculated Mean Value** | 15.18182 | 40.72727 | 58.61364 |
| **Standard Value** | < 49 | <= 78.34 |  |



Graph 2- Libsu- Master -Metrics for Maintainability (complete analysis)

The above graph is generated for Libsu-Master which considers maintainability. It shows the complete values for WMC, LCOM, and LOC.

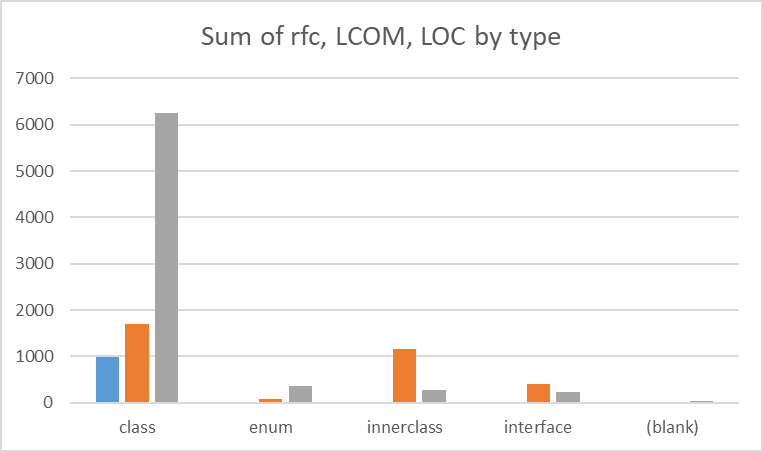


Graph 3-Libsu- Master -Metrics for Maintainability

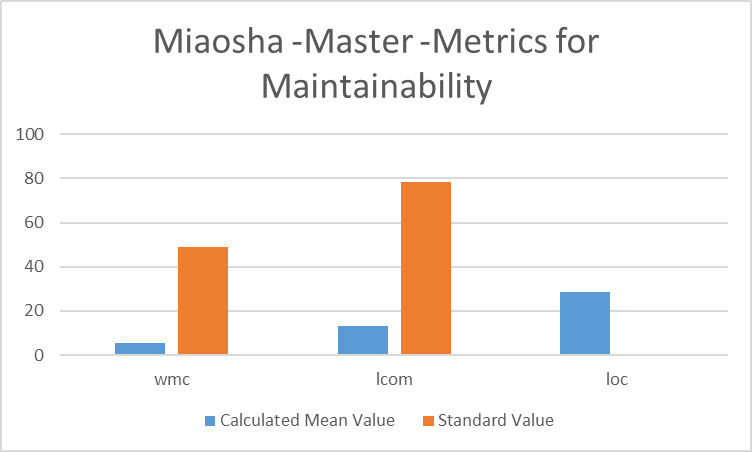
The above graph shows the comparison between the calculated average value and the standard value.

Table 4-Miaosha -Master -Metrics for Maintainability

|  |  |  |  |
| --- | --- | --- | --- |
|  | **WMC** | **lcom** | **loc** |
| **Calculated Mean Value** | 5.62 | 13.28 | 28.468 |
| **Standard Value** | < 49 | <= 78.34 |  |



Graph 4- Miaosha -Master -Metrics for Maintainability (Sum values by type)

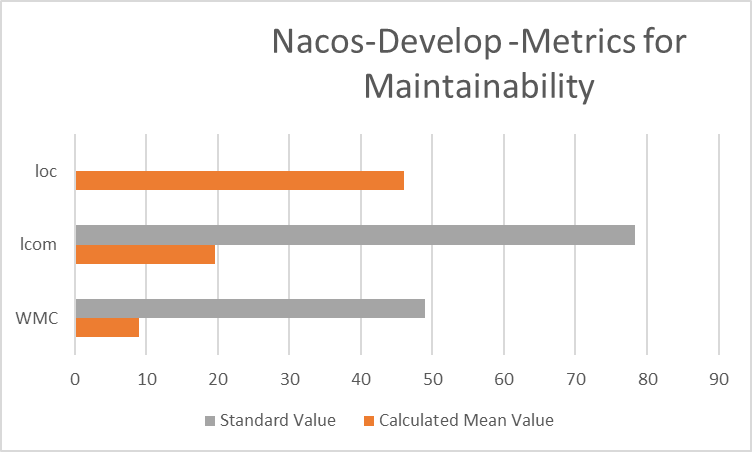


Graph 5-Miaosha -Master -Metrics for Maintainability

The above table and graph show the “Miaosha master value” project and its metrics value for measuring maintainability. When calculating the average value it shows lower when compared to the standard value. Because it is an average value, but the individual class value is higher so the developers should consider and maintain the standard value.

Table 5 -Nacos-Develop -Metrics for Maintainability

|  |  |  |  |
| --- | --- | --- | --- |
|  | **WMC** | **lcom** | **loc** |
| **Calculated Mean Value** | 8.926122 | 19.61714 | 46.06286 |
| **Standard Value** | < 49 | < =78.34 |  |

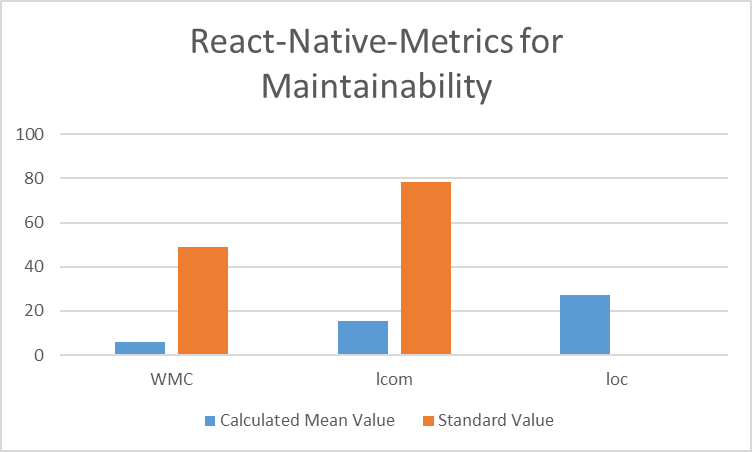


Graph 6-Nacos-Develop -Metrics for Maintainability

The graph highlights the standard value and calculated average value for “Nacos-Develop”.

Table 6-React-Native-Metrics for Maintainability

|  |  |  |  |
| --- | --- | --- | --- |
|  | **WMC** | **lcom** | **loc** |
| **Calculated Mean Value** | 5.92233 | 15.3301 | 26.96117 |
| **Standard Value** | < 49 | < =78.34 |  |

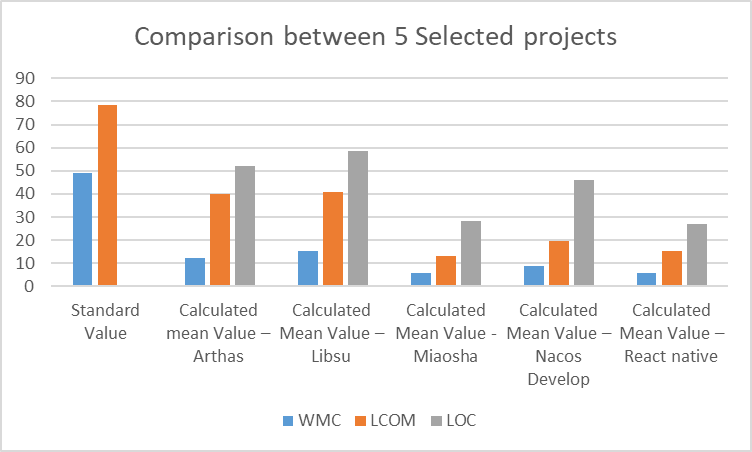


Graph 7-React-Native-Metrics for Maintainability

The graph highlights the maintainability metrics which is selected for the project of “React-Native.”

Table 7- Comparisons between different projects

|  |  |  |  |
| --- | --- | --- | --- |
|  | **WMC** | **LCOM** | **LOC** |
| **Standard Value** | < 49 | < = 78.34 |  |
| **Calculated mean Value – Arthas** | 12.19716 | 39.85954 | 51.97036 |
| **Calculated Mean Value – Libsu** | 15.1818 | 40.72727 | 58.61364 |
| **Calculated Mean Value -Miaosha** | 5.62 | 13.28 | 28.468 |
| **Calculated Mean Value –Nacos Develop** | 8.926122 | 19.61714 | 46.06286 |
| **Calculated Mean Value – React Native** | 5.92233 | 15.3301 | 26.96117 |



Graph 8-Comparisons between different projects

The above table and graph show the comparison of the selected CK metrics values for the selected five projects. When comparing all the projects, “Measho” has lesser WMC and LCOM values. So maintainability for this code is higher when compared to other project works [7]. Every project code is calculated with LOC value which is essential to measure the class size. There is no standard LOC value but a lesser LOC value ensures higher maintainability. When comparing all five projects, “React-Native” has lower LOC values.

# Section 6

## Conclusion

The proposed empirical research work focuses on two CK metrics values such as WMC and LCOM values for measuring the class size and maintainability. WMC value greatly affects the class size which leads to maintainability issues. Size is commonly referred to as the source lines of code (SLOC). It majorly affects the maintainability of the code. The method SLOC is positively correlated with maintenance effort so developers should minimize the LOC value. In conclusion, the investigation of CK metrics (LOC, LCOM, and WMC) in relation to class size has provided valuable insights into the effect of class size on software maintainability.

The findings show that class size does have an impact on maintainability. It shows that larger class sizes result in larger LOC values that indicate code complexity and are harder to maintain. The developers should consider the smaller LOC values to improve maintainability. Larger class size increases WMC metrics values so it is difficult to understand the code. Smaller code sizes ensure code readability and maintainability.

Overall, the findings from investigating CK metrics in relation to class size suggest that smaller class sizes tend to lead to better software maintainability. Reducing class size can result in less complex code, higher cohesion, and simpler methods, which can improve understandability, maintainability, testability, and software debugging. Many contributors involved in the software code development who requires an effective maintainability that means code should flexible to add new features. To ensure software quality and maintainability, developers should consider the class size. When analyzing the five different software codes, it shows that need to consider the CK metric standard values to achieve higher maintainability and software quality.

# References

|  |  |
| --- | --- |
| [1] | S. A. Chowdhury, G. Uddin and R. Holmes, “An Empirical Study on Maintainable Method Size in Java,” 2022. |
| [2] | Z. Wang and D. Zhang, “Improved CK Metrics for Object Oriented Design Measurement,” *Advanced Engineering Forum,* vol. 6, no. 7, pp. 333-336, 2012. |
| [3] | B. Seref and O. Tanriover, “Software Code Maintainability : A Literature Review,” *International Journal of Software Engineering & Applications (IJSEA),* vol. 7, no. 3, pp. 69-87, 2016. |
| [4] | N. Qamar and A. A. Malik, “Impact of Design Patterns on Software Complexity and Size,” *Mehran University Research Journal of Engineering and Technology,* vol. 39, no. 2, pp. 342-352, 2020. |
| [5] | O. Jarczyk, B. Gruszka, L. Bukowski and S. Jaroszewicz, “GitHub Projects. Quality Analysis of Open-Source Software,” *International Conference on Social Informatics,* pp. 80-94, 2014. |
| [6] | S. Amann, S. Proksch, S. Nadi and M. Mezini, “A Study of Visual Studio Usage in Practice,” *IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER),* 2016. |
| [7] | Z. Q. Wang and D. K. Zhang, “Improved CK Metrics for Object Oriented Design Measurement,” *Advanced Engineering Forum ,* vol. 6, no. 7, pp. 333-336, 2012. |
| [8] | R. Subramanyam and M. S. Krishnan, “Empirical Analysis of CK Metrics for Object-Oriented Design Complexity: Implications for Software Defects,” *IEEE Transactions on Software Engineering ,* vol. 29, no. 4, pp. 297-310, 2003. |