# **Group Assignment 1**

# FINAL REPORT

By:  
 Abhivardhan Tammana(L30086385)

Suprathika Vajrala

# Objectives, Questions, and Metrics

**Objective:** Analyze the relationship between code size and maintainability in Java projects.

**Question:** How does the size (lines of code) of a class impact its potential for interactions (measured by Response Sets for a Class) (measured by other CK metrics)?

**Metrics:**

Class Size **(LoC):** Lines of code for each class.

Response Sets for a Class **(RFC):** Set of methods potentially executed in response to a message sent to an object of the class.

Description of Subject Programs (Data Set)  
  
For this study, we choose 5 Java projects from GitHub as our subject programs. We choose the following projects based on these criteria to ensure the relevance and suitability of the projects:  
  
**Size Criteria:** We made sure that each selected project has a minimum size of 10,000 lines of code (LoC).  
  
**Age Criteria:** We included projects that are at least 3 years old.

**Development Team Size:** Each project must have been developed by a team of at least 3 developers.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.No | Project Name | Size (LOC) | Age(years) | Number of Developers | Description |
| 1 | alibaba/druid | 65997 | 12+ | 169 | It's an open-source, column-oriented, distributed data store written in Java. It's designed to quickly ingest large volumes of event data and provide low-latency queries on that data. |
| 2 | baomidou/mybatis-plus | 17145 | 7+ | 121 | Baomidou's MyBatis-Plus is a powerful extension library for MyBatis in Java, offering features like code generation, fluent query building, and built-in pagination to simplify database operations and enhance productivity |
| 3 | alibaba/DataX | 19714 | 5+ | 52 | Alibaba's DataX is an open-source data synchronization tool designed for efficiently transferring data between various sources and destinations. It supports a wide range of data formats and storage systems, offering scalability, reliability, and flexibility for data integration tasks. |
| 4 | zaproxy/zaproxy | 191285 | 8+ | 180 | is an open-source web application security testing tool designed to help identify vulnerabilities in web applications during development and testing phases. It offers features such as automated scanning, manual testing, and passive scanning to detect and report security issues, making it a valuable tool for developers and security professionals alike. |
| 5 | metersphere/metersphere | 116813 | 3+ | 47 | MeterSphere is a one-stop open-source enterprise-class continuous testing platform. It covers functions such as tests tracking, interface testing, performance testing, team collaboration and is compatible with open-source standards such as JMeter. |

Description of the Tool Used:

We utilized a well-established software metric suite to analyze the projects in our study. While the suite offers various metrics, we focused on two key indicators: Response For Class (RFC) and Lines of Code (LOC).

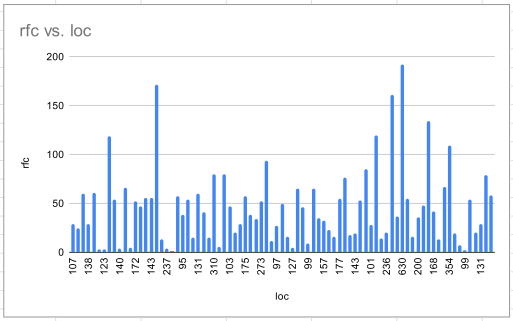
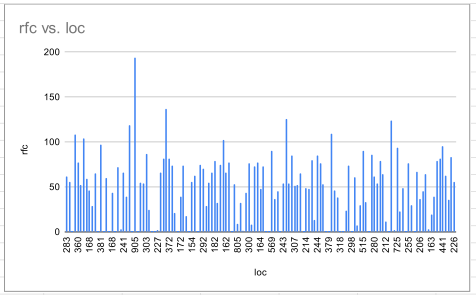
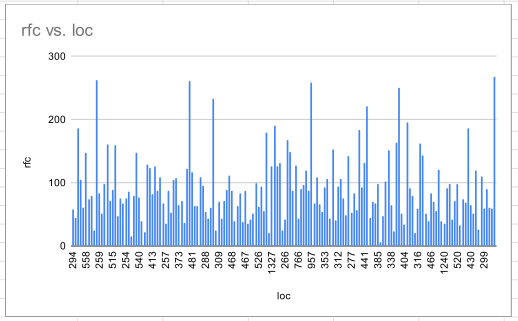
**Metrics Utilized:**Response For Class (RFC): This metric captures the number of methods a class either implements directly or inherits. High RFC values can indicate complex classes with potential maintainability challenges.

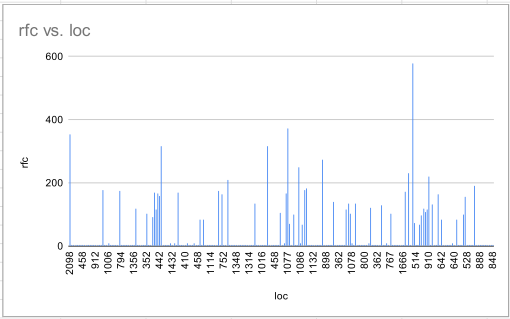
Lines of Code (LOC): LOC provides a basic measure of class size. Though not a sole indicator, it can be helpful when analyzed alongside RFC to understand overall class complexity.  
  
**Tool Usage:**  
The CK-Code metrics for Java code: <https://github.com/mauricioaniche/ck>.  
  
  
  
  
  
  
  
Reporting Results:  
The analysis of the following Java projects using RFC and LoC metrics reveals a correlation between class size and complexity, highlighting potential maintenance challenges in classes with higher RFC-to-LoC ratios.  
  
RFC-LoC Correlation: Larger classes (higher LOC) tend to be more complex (higher RFC).  
  
Maintenance Challenges: High RFC-to-LoC classes pose maintenance challenges.  
  
High-Complexity Class Identification: We identified potential complexity hotspots by examining classes with the highest RFC relative to their LOC. These classes warrant closer attention during maintenance.

alibaba/druid  
 A graph with blue lines

Description automatically generated  
 Figure 1: alibaba/druid LOC vs RFC

baomidou/mybatis-plus

  
 Figure 2: baomidou/mybatis-plus LOC vs RFC  
  
  
alibaba/DataX  
   
 Figure 3: alibaba/DataX LOC vs RFC  
  
  
  
zaproxy/zaproxy  
  
Figure 4: zaproxy/zaproxy LOC vs RFC

metersphere/metersphere  
  
Figure 5: metersphere/metersphere LOC vs RFC

CONCLUSION

Our analysis of the Java projects, focusing on metrics like RFC (Response for a Class) and LoC (Lines of Code), yields significant findings:

Impact of Size on Complexity: Larger classes, as measured by LoC, often exhibit higher complexity, as indicated by RFC. This correlation suggests that class size contributes to code complexity, affecting maintainability.

Maintenance Challenges: Classes with higher RFC values may pose greater maintenance challenges, requiring more effort to understand and modify. This underscores the importance of managing class complexity to enhance maintainability.

Efficiency of Codebase: Projects with lower RFC-to-LoC ratios may indicate more efficient and maintainable codebases, where classes are more focused and have fewer responsibilities.

References

* CK-Code metrics for Java code. Available at: [<https://github.com/mauricioaniche/ck>]
* <https://github.com/alibaba/druid>
* <https://github.com/baomidou/mybatis-plus>
* <https://github.com/alibaba/DataX>
* <https://github.com/zaproxy/zaproxy>
* <https://github.com/metersphere/metersphere>