ASSIGNMENT 1

FINAL REPORT

By:

Meghana Priya Lakamana

Navadeep Gottigundala

Date: May 2024

# Contents

1. [Objectives, Questions, and Metrics](#_bookmark0) 1
   1. [Objectives:](#_bookmark1) 1
   2. [Questions:](#_bookmark2) 1
   3. [Metrics:](#_bookmark3) 1
2. [Description of Subject Programs (Data Set)](#_bookmark4) 2
3. [Description of the Tool Used:](#_bookmark6) 3
4. [Reporting Results](#_bookmark7) 3
5. [Conclusion](#_bookmark13) 6
6. [References](#_bookmark14) 6

List of Tables

[2 Project Details.](#_bookmark5) 2

List of Figures

1. [airbyte LOC vs DIT](#_bookmark8) 3
2. [compreface LOC vs DIT](#_bookmark9) 4
3. [metersphere LOC vs DIT](#_bookmark10) 4
4. [pojavlaunch LOC vs DIT](#_bookmark11) 5
5. [smarttube LOC vs DIT](#_bookmark12) 5

# Objectives, Questions, and Metrics

## Objectives:

* + - Finding a relationship between a Java project's scale and maintainability is the aim of the current study.
    - This task's primary goal is to identify unusual numerical values and classify related items in the data collection.
    - The properties and implications of the metrics associated with the CK of Java projects will be covered in detail and scientifically in this chapter.

## Questions:

* + - What is the relationship between the number of lines for a particular project and the metrics on the maintainability (eg., dıt)?
    - Perhaps you see trends or shapes in the data, informing about the connection of the project size and maintenance.
    - Is there any influencer or community whose result elicits surprise or whose discovery the data will produce?

## Metrics:

* + - The baselines used in the project (loc = number of lines of code).
    - Maintainability metrics (e.g., ’dit’)

# Description of Subject Programs (Data Set)

We selected 5 Java projects from GitHub based on the following criteria:

* Over 10K lines of code program.
* Projects that either last 3 years and more.
* Project with a team comprising of at least three devs engaged.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| S.No | Project Name | Size (LOC) | Age(years) | Number of Developers | Description |
| 1 | airbytehq/airbyte | 215523 | 4 | 355 | Airbyte is  an open- source EL(T)  platform that helps you repli-  cate your data . |
| 2 | exadel-inc/CompreFace | 512922 | 4 | 64 | Leading  free and open- source face recognition system |
| 3 | metersphere/metersphere | 116813 | 4 | 47 | MeterSphere |
| 4 | PojavLauncherTeam/PojavLauncher | 717939 | 4 | 14 | A  Minecraft: Java Edition Launcher for An-  droid and iOS based on Board- walk. |
| 5 | yuliskov/SmartTubeNext | 62356 | 4 | 45 | Ad free  app for watching tube videos on Android TV boxes |

Table 2: Project Details.

# Description of the Tool Used:

We decided to use the CK-Code metrics tool since it was created by a team of 24 developers with an emphasis on Java applications. The ReadMe file's instructions were modified to enable the code fragment to be viewed from GitHub.

Tool Citation: Performance analysis for CK-Code with their metrics for Java code.Available at: [https://github.com/mauricio

# Reporting Results

* The relationship between ’dit’ and ’loc’ values was appropriately illustrated throughout all projects by means of bar charts.
* Analysis of acquired measurements allowed us to get the picture concerning Project size distribution, to search for outliers, and to compare clusters of projects.
* You will find the graphs attached to help you see the results more clearly.

A graph with red bars

Description automatically generated

Figure 1: airbyte LOC vs DIT

A graph with red bars

Description automatically generated

Figure 2: compreface LOC vs DIT

A graph with red lines

Description automatically generated

Figure 3: metersphere LOC vs DIT

A graph with numbers and lines

Description automatically generated

Figure 4: pojavlaunch LOC vs DIT

A graph with red lines

Description automatically generated

Figure 5: smarttube LOC vs DIT

# Conclusion

Based on the analysis of the obtained results:Based on the analysis of the obtained results:

* The number of lines of code (LOC), which is a common indicator of a project's size, and its complexity, which is frequently calculated as "dat," seem to be directly correlated.
* The data exhibits a significant variance in project sizes, indicating that projects vary greatly in size.
* The identified groups and outliers can be very helpful when it comes to the characteristics unique to the project and the factors that affect the size diversity.
* The analysis's conclusions will assist readers in comprehending the steps involved in Java projects as a whole and will provide information on potential maintainability problems that could influence their choices.

# References

1. CK-Code metrics for Java code. Available at: [https://github.com/mauricioaniche/ck]
2. <https://github.com/airbytehq/airbyte>
3. <https://github.com/exadel-inc/CompreFace>
4. <https://github.com/metersphere/metersphere>
5. <https://github.com/PojavLauncherTeam/PojavLauncher>
6. <https://github.com/yuliskov/SmartTube>
7. Basili, V. R., Briand, L. C., & Melo, W. L. (1996). "A validation of object-oriented design metrics as quality indicators." IEEE Transactions on Software Engineering, 22(10), 751-761.
8. Chidamber, S. R., & Kemerer, C. F. (1994). "A metrics suite for object oriented design." IEEE Transactions on Software Engineering, 20(6), 476-493.
9. Cohen, J. (1988). "Statistical power analysis for the behavioral sciences" (2nd ed.). Hillsdale, NJ: Lawrence Earlbaum Associates.