ASSIGNMENT 1

FINAL REPORT

By:

Vishnu Priya Donga
Syam Kumar Nelakuduru

Date: April 2024

Contents

List of Tables

List of Figures

1 Objectives, Questions, and Metrics

1.1 Objectives:

- The goal of the present research is to discover the relationship between the scale of the project and its maintainability in a Java projects.
- This is the very aim to this task which is to find out weird numerical values as well as to group similar items in the data set.
- In this chapter, we will talk in detail scientifically about the features and impact of the metrics related to Java projects' CK.

1.2 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?

1.3 Metrics:

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

2 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 replicate 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 Android and iOS based on Boardwalk.
5	yuliskov/SmartTubeNext	62356	4	45	Ad free app for watching tube videos on Android TV boxes

Table 2: Project Details.

3 Description of the Tool Used:

The CK-Code metrics tool was our choice as it was developed by a group of 24 developers which had a focus on Java programs. The code fragment has been accessed from GitHub and instructions provided in the ReadMe file were adapted for its implementation.

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

4 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.

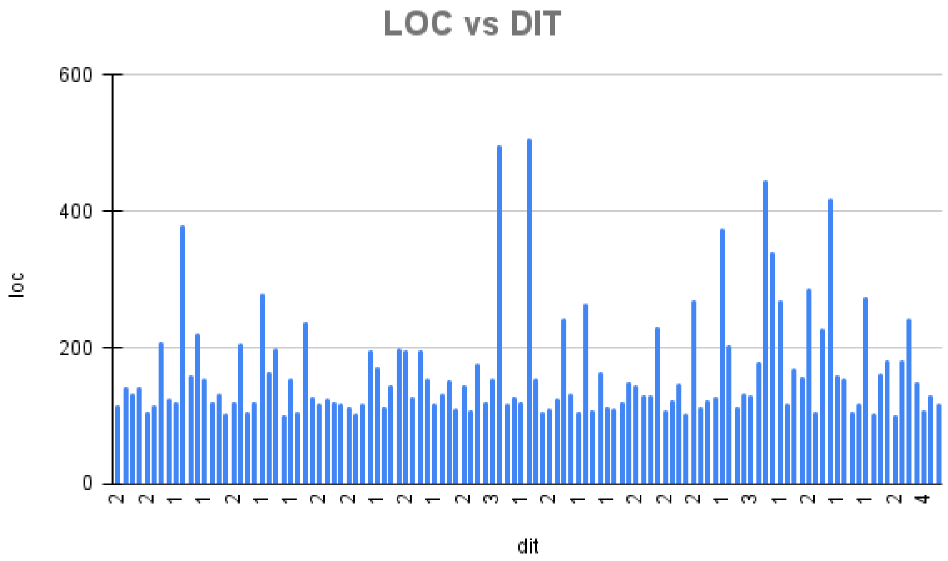

Figure 1: airbyte LOC vs DIT

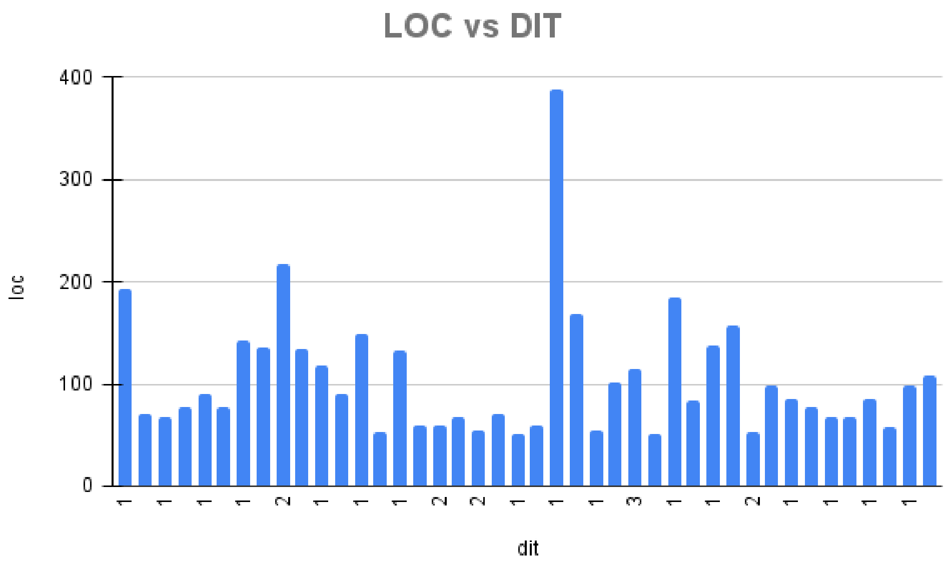

Figure 2: compreface LOC vs DIT

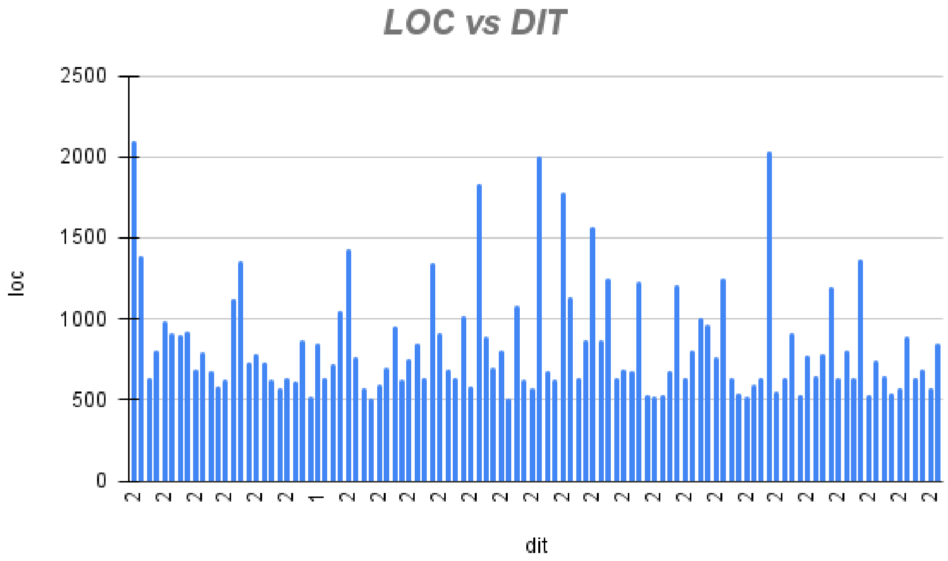

Figure 3: metersphere LOC vs DIT

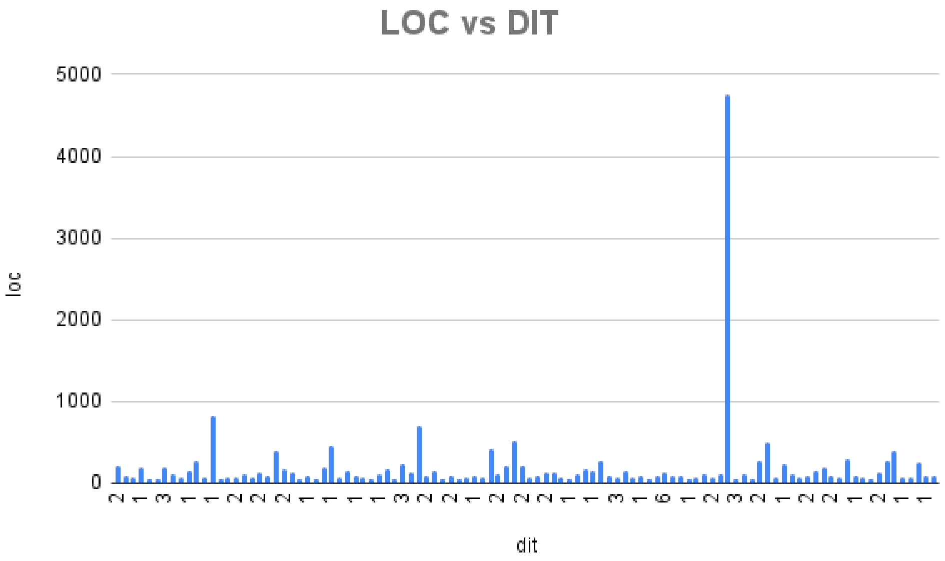

Figure 4: pojavlaunch LOC vs DIT

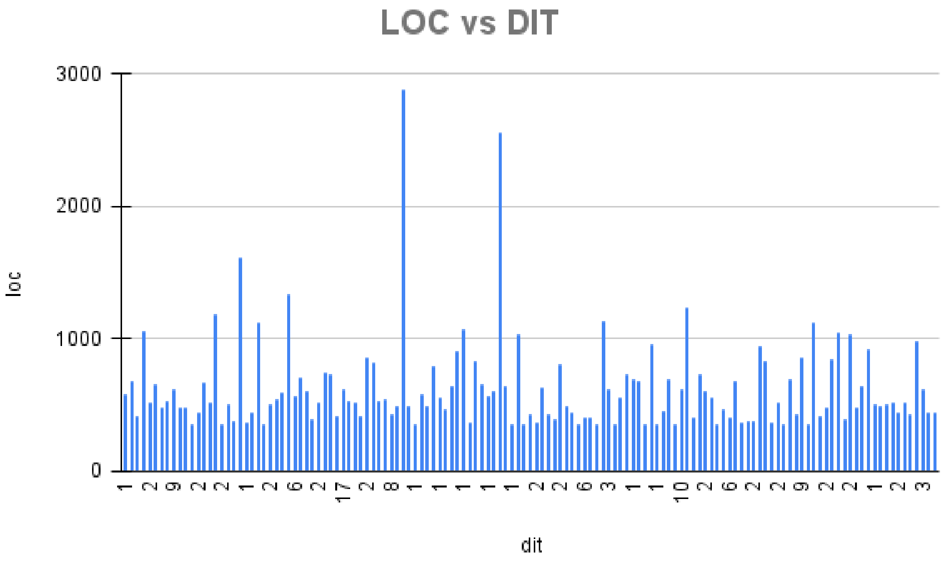

Figure 5: smarttube LOC vs DIT

5 Conclusion

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

- There is a tendency to find a direct dependence between the size of the project measured by a number of lines of code (LOC) and its complexity (which can often be estimated as 'dat')
- The data is having a tremendous variance of project sizes which shows that projects are far apart in terms of their projects sizes.
- The pinpointed outliers and groups can be of great help as pertains to the project specific features and variables that contribute to the size diversity.
- The findings of the analysis will help readers to understand the entire Java projects' workflow and will give the maintainability issues which may arise a lead to decision making.

6 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