

Developer Productivity Analytics Using Git Commit Data

MUTHU KARTHIK V

Department of Computer Science and Engineering SRG Engineering College, Namakkal, Tamil Nadu

PRAJWAL R

Department of Computer Science and Engineering SRG Engineering College, Namakkal, Tamil Nadu

YUVARAJ S

Department of Computer Science and Engineering SRG Engineering College, Namakkal, Tamil Nadu

DEEPIKA L

Assistant Professor

Department of Computer Science and Engineering SRG Engineering College, Namakkal, Tamil Nadu

ABSTRACT

Software development teams rely heavily on version control systems such as Git to manage and track changes in source code. Every code modification performed by developers is stored as commit data, which contains valuable information regarding development activity and contribution patterns. However, many organizations do not fully utilize this data to evaluate developer productivity. Traditional evaluation methods are often manual and subjective, which may lead to inaccurate performance assessments.

This paper proposes a Developer Productivity Analytics system that analyzes Git commit data to evaluate developer contributions. The system extracts key metrics such as commit frequency, lines of code added or deleted, active development days, and code churn. These metrics are processed using data analytics techniques and visualized through dashboards and reports. The proposed system transforms raw Git repository data into meaningful productivity insights that support data-driven decision-making and transparent performance evaluation in software development teams.

Keywords: Developer Productivity, Git Analytics, Software Engineering Metrics, Data Analysis, Version Control Systems

I. INTRODUCTION

Modern software development relies heavily on collaborative tools and version control systems to manage project workflows.

Git has become the most widely used distributed version control system for managing source code. Every change made to the software is recorded in the form of commits, which include details such as author information, timestamp, commit message, and code modifications.

Despite the availability of this rich data, many organizations still rely on manual or subjective methods to evaluate developer productivity. These approaches often depend on managerial opinions rather than measurable indicators, which may lead to inconsistent or biased evaluations.

Git repositories contain valuable data that can be used to understand development behavior and productivity patterns. By analyzing commit logs and related information, it is possible to derive meaningful metrics that reflect developer contributions. Data analytics techniques can transform this raw repository data into structured productivity insights.

II. LITERATURE SURVEY

Several studies have explored the use of software repositories to analyze developer productivity and software quality.

Version control systems such as Git provide large datasets that contain information about developer activities and project evolution.

Bird et al. introduced repository mining techniques to analyze development patterns and collaboration behavior within software projects. Their work demonstrated that repository data can be used to identify developer contributions and project trends.

Mockus and Herbsleb studied open-source software development and found that commit activity and contribution frequency provide useful insights into developer productivity. However, their research also emphasized the need for better analytical tools to interpret repository data effectively.

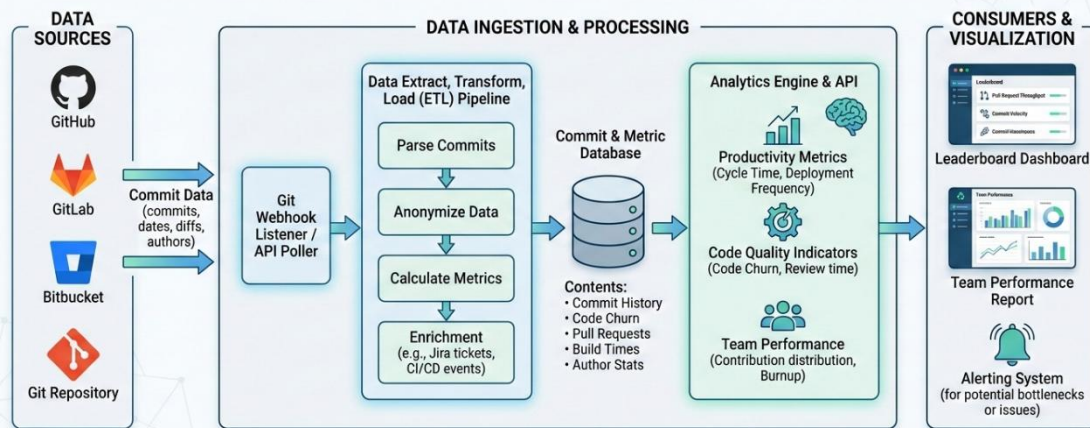
III. PROPOSED SYSTEM

The proposed system introduces an automated analytics framework that evaluates developer productivity using Git commit data. The system extracts commit logs from Git repositories and processes them to generate various productivity metrics.

The architecture consists of multiple modules including data extraction, data processing, analytics computation, and visualization. The system collects commit information such as commit author, timestamp, and lines of code changed.

These data points are analyzed to compute productivity indicators including commit count, code contribution levels, developer activity frequency, and code churn metrics. Visualization tools generate graphical reports that help managers interpret developer performance.

SYSTEM ARCHITECTURE FOR DEVELOPER PRODUCTIVITY ANALYTICS USING GIT COMMIT DATA



IV. METHODOLOGY

4.1 Data Collection

Commit data is extracted from Git repositories using Git commands or APIs. The extracted information includes commit author, commit timestamp, and lines of code added or deleted.

4.2 Data Preprocessing

The collected commit data is cleaned and structured using data processing techniques. Duplicate entries and irrelevant data are removed to ensure accurate analysis.

4.3 Metric Calculation

Several productivity metrics are calculated:

- Number of commits
- Lines of code added
- Lines of code deleted
- Active development days
- Code churn rate

4.4 Visualization

The processed data is visualized using charts and graphs to represent productivity trends and contribution patterns.

4.5 Productivity Analysis

The final stage analyzes metrics to identify developer performance trends and project activity patterns.

V. RESULT AND DISCUSSION

The proposed system was evaluated using Git repositories containing commit history data. The analysis generated productivity metrics and visual reports illustrating developer activity patterns.

The results indicate that commit frequency and code churn metrics provide useful indicators of developer activity. Visualization dashboards enable project managers to easily compare developer contributions and identify productivity trends.

Compared to traditional manual evaluation methods, the proposed system provides a transparent and data-driven approach for measuring developer productivity.

VI. CONCLUSION

This paper presented a Developer Productivity Analytics system that analyzes Git commit data to evaluate developer contributions. The system extracts commit logs from Git repositories and processes them using data analytics techniques to generate meaningful productivity insights.

The approach provides objective productivity measurement and improves transparency in performance evaluation. Future work may include integrating machine learning techniques and task management data to enhance developer productivity analysis.

REFERENCES

- [1] T. Bird et al., Mining Software Repositories, IEEE Software Engineering.
- [2] A. Mockus and J. Herbsleb, Open Source Development Studies, ACM.
- [3] Git Documentation – <https://git-scm.com/docs>
- [4] Python Documentation – <https://docs.python.org>