Software Component Time-Domain Visualization

David Kavaler, Sugeerth Murugesan, Vishak Muthukumar

Project Description and Motivation

There are well-established metrics that are indicators of potential bugs in software.

However, conceptualizing these metrics in a dynamic, evolving system is difficult.

We aim to provide a visualization tool that can be used to identify fault-prone code using established metrics that can change value over time.

Project Progress

- Data mining
  - Git history of Apache Software Foundation projects
  - Metrics (per file per commit) based on Git history
  - Bug data based on JIRA issues
  - CVE data
- Prototype visualization tool

Apache Software Foundation Git Data Mining

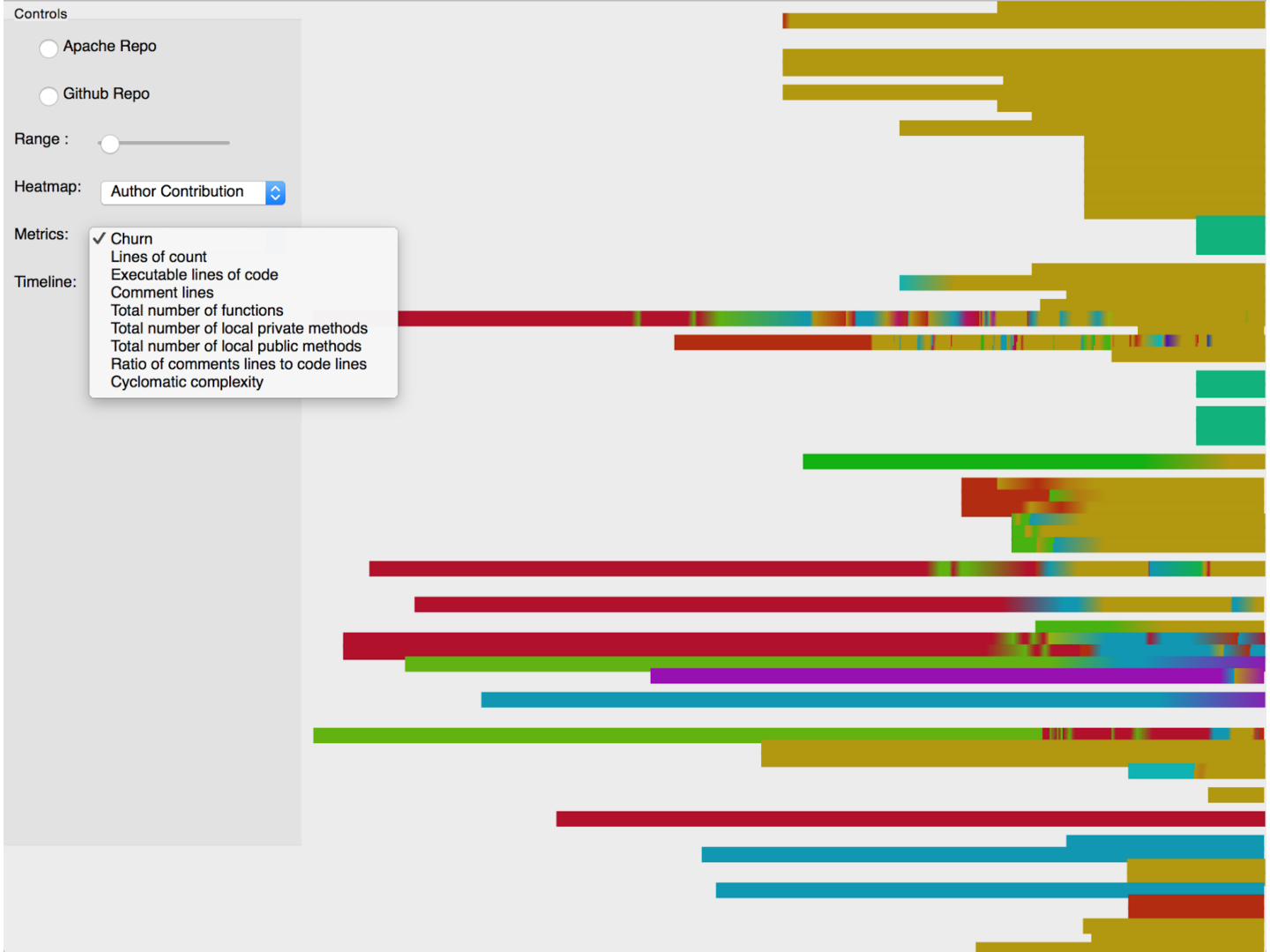
- Clone each repository
- For each repository, extract commit dates, authoring dates, lines added and deleted, committer and author IDs according to Git history
- Extract software metrics for every revision

Gathered Metrics

- Count metrics
  - Total count LOC
  - Count of executable LOC
  - Comment LOC
  - Churn (added and deleted LOC)
- Object-oriented feature metrics
  - Total number of functions
  - Total number of local public methods
  - Total number of local private methods
- Complexity
  - Ratio of comment lines to code lines
  - Sum of cyclomatic complexity
- Misc.
  - Authors and committers

Bug Extraction Process

- Apache Software Foundation projects use JIRA to track issues
- Each addressed JIRA issue is linked to a fixing commit by process
- Run git blame to find which revision(s) last modified the lines in the fixing commit
- Provides issue-fixing and issue-introducing commits for each issue, allowing us to track each issue’s history in each repository


Controls
Apache Repo
Github Repo
Range :
Heatmap:
Author Contribution
Metrics:
Churn
Lines of count
Executable lines of code
Comment lines
Total number of functions
Total number of local private methods
Total number of local public methods
Ratio of comments lines to code lines
Cyclomatic complexity
Timeline: