Sprint 2 Design Document

EMMA BENNETT, STELLA CRAIG, ANDERS RYAN, BASHEER ABBAS SHAIK

Table of Contents

1 Introduction

Open-source projects have been critical throughout time in the development of unique, efficient solutions to a wide-ranging spectrum of computing problems. Broadly speaking, open-source software is software in which the copyright licensure allows users to modify or contribute to the underlying code in collaboration with other users and developers. Users can adapt software to personal needs, collaborate to produce updates, fix bugs, etc.

The health and sustainability of these open-source projects requires analyzing a multitude of different contribution statistics for a project over time. The ability to use these statistics to make meaningful conclusions about the project is hugely invaluable. Having current knowledge of a project's health and the ability to predict how contribution patterns will grow or change allow companies and organizations to concentrate resources, allows contributors to see the big picture results of their efforts, and provides tools for attracting new members and promoting the project.

The organization Community Health Analytics in Open-Source Software (CHAOSS) specifically develops useful health metrics and health metric models to allow for the assessment of the health of open-source projects and as well as provides a community for developers interested in advancing health metric tracking.

2 Metrics Overview

Selected for development in this project are the addition of two metric models to the existing 8Knot system. These models, as described below, will track useful project metrics which provide quantifiable feedback about project health and stability to the key actors that interact with this system.

2.1 Starter Project Health Metrics Model

The first metric model selected for addition to this system is the Starter Project Health Metrics Model. The information that builds this model, per CHAOSS, provides a simplified version of the most critical components related to project health. This model is intended to be a starting point, the first step an open-source project can take in furthering their understanding of the risk points and overall health of their project. These models are simple to understand, and generally applicable to most open-source projects. They include bus factor, time to first response, change request closure ratio, and release frequency (Metrics and Metric Models, 2023). These metrics are described in detail below.

2.1.1 Bus Factor

The bus factor metric represents the number of people that could feasibly leave a project without dropping the project contribution rate significantly. Bus factor is based on the number of contributors that make up 50% of contributions. This is a critical metric for project health and of particular interest to project stakeholders, as if a project is being carried by a single, or small amount, of contributors, it is at high risk for failure if those individuals stop contributing.

2.1.2 Time to First Response

Time to first response is a metric which tracks the average time a contributor's pull request sits before receiving interaction from a real-person maintainer for the project. The longer on average it takes for these requests to receive responses, the more at risk a project becomes as contributors can become discouraged if feedback time is slow. A general target for most projects is to stay within a ~2-day response window. This metric is in particular very helpful for project maintainers so that they can ensure adequate response speed.

2.1.3 Change Request Closure Ratio

Like time to first response, change request closure ratio is an important metric for project maintainers to track to make sure they don't become far behind on response times to their contributors. Change request closure ratio tracks how many total change requests have been merged or closed without merge. A higher deficit between total requests and closed requests indicates that the maintainers of the project are starting to lag, indicating a risk for the project.

2.1.4 Release Frequency

The goal of any open-source project is to release new updates and continue improving software. The frequency for these releases, anything from bug-fixes to new features, is an important metric to track to verify the project is seeing progress. Projects with low release rates are at risk for stagnation, and having a consistent release schedule indicates a healthy project. All releases, big and small, are accounted for in this metric. This metric also provides critical information to stakeholders, particularly those that may not be as technically fluent, about the robustness of how the project is progressing.

3 Starter Health Metrics Design

The following section describes further detail about the architectural and detailed design components of the Starter Health Metric Model's implementation in Sprint 2 of this effort. As this is an additional feature to a preexisting software with frontend and backend design, the focus of this section is describing how the backend database will be queried/accessed and how the existing fronted stack will be utilized to build intuitive visualizations which display each metric.

3.1 Bus Factor - Stella

3.1.1 Database Querying

To get the appropriate content to construct the bus factor graph the materialized view "augur_data.explorer_contributor_actions" was queried for repository ID, repository name, contributor ID, created at, the login, action, and rank. This query was done through an additional queries python file which was added to the index callback function under the tag "ctq". The bus factor file was then used to generate the appropriate data frame.

3.1.2 Metric Visualization

The visualization chosen for this metric was a pie chart which displays the top contributors by commit. This was selected as it provides a clear visual of the minimum number of contributors that made up a certain percentage of commits. The contributors shown are the contributors that made up the largest percentages of commits. The number of contributors on display can also be edited through a text box below the graph. Additionally, for each pie slice the contributor id and number of contributions will be displayed to the exact number if hovered over.

3.2 Time to First Response - Emma

3.2.1 Database Querying

To get the appropriate content to construct the time to first response graph the materialized view "explorer_pr_response_times" was queried for repository ID, repository name, hours to first response, and pull request closed. This query was done through an additional queries python file which was added to the index callback function under the tag "rtq". The time to first response visualization file was then used to generate the appropriate data frame.

3.2.2 Metric Visualization

The visualization chosen for this metric was a bar graph which displays, in mean hours, the time to first response for an individual pull request for a defined repository. This was selected as it provides a clear visual of any dates which the mean response time was particularly high, it also allows for multiple dates to be compared over a specified time range. As clarified in the metric description, the target response time is 48 hours (or 2 days). Pull requests are organized by the date on which they were closed. Mean first response times can be organized by day, week, month, and year through interactive fields. The date range on display can also be edited through a draggable frame below the graph. Additionally, for each bar the date and mean response time in hours will be displayed to the exact number if hovered over.

3.3 Change Request Closure Ratio - Basheer

3.3.1 Database Querying

3.3.2 Metric Visualization

3.4 Release Frequency - Anders

3.4.1 Database Querying

To retrieve the correct data for the visualization the "releases" table was queried for repo_id, release_id and release_published_at by a selected repo_id. This data was then used to count the total number of releases over a certain period. This query was created in a separate release_frequecny_query.py file and then used in the release_frequency.py file to display the queried data in a data frame as a bar graph.

3.4.2 Metric Visualization

The visualization for the release frequency was chosen to be a bar graph that shows the total number of releases in a certain time interval selected by the user. The user can either select the time intervals of day, week, month to year for the data to be shown over. A bar graph was selected to visualize the data because it provides a clear and effective way to display the number of releases over certain periods of time. The user is also given the ability to select the total date range of the data via a draggable frame below the bar graph. This allows the user to see how the number of releases has changed over a longer period of time. When each bar of the bar graph is hovered over with the mouse it will display the exact period (day, week, month year) and number of releases that bar represents.

4 Works Cited

(2023, Sept 18). Retrieved from Augur Documentation: https://oss-augur.readthedocs.io/en/main/index.html

(2023, Sept 18). Retrieved from chaoss / augur: https://github.com/chaoss/augur

Metrics and Metric Models. (2023). Retrieved from CHAOSS: https://chaoss.community/kb-metrics-and-metrics-models/