Sprint 1 Requirements 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.

2.2 Collaboration Development Index Metrics Model

The goal of this metric model is to provide a quantifiable way to track the community environment surrounding an open-source project. Open-source projects are essentially representative of the collective intelligence of a group or community, and how that community works together and collaborates can make or break a project. The metrics within this model include contributors, downloads, code change commits, CI test, code commit linked with change request, ratio of issues to change requests, technical forks, and code change lines (Metrics and Metric Models, 2023). Only five, listed in this section, have been selected from this model for implementation as part of this effort. They are described in detail below.

2.2.1 Contributors

The contributors metric tracks the number of users (change requests, commit authors, reviewers) that have been active on the project within the past 90 days. This can be useful to track to see the ebbs and flows of a project and determine the overall trend. Projects which are trending downward in contributor activity may be at higher risk.

2.2.2 Code Change Commits

Code change commits track, per 90-day periods, the number of commits per week, the percentage of weeks with at least code commit, and the percentage of repositories with at least one commit (Metrics and Metric Models, 2023). This is a useful metric to analyze how active the coding community is on a project. Projects with waning activity are potentially at risk. This metric provides information similar to the contributors metric in that it seeks to quantify project activity, but it is more tailor focused on commits to the repositories themselves than the individual user interaction with the project (i.e. comments, reviews).

2.2.3 Change Request Reviews

This metric tracks the procedures for which change requests are reviewed and processed on a project. This can relate to the quality of review being completed on the change request (i.e. does the project have a formal contribution procedures document, if not do they need one). The metric provides analysis of how many change requests were reviewed, accepted, commented on and the number of contributors providing reviews versus authoring change requests over a 90-day period. The percentage of change requests who had at least one non-author reviewer are also tracked. This metric can be useful, for example if a project becomes bogged down with many rejected change requests. This puts a project at risk for stagnation as the deficit grows between when contributors submit a request and when maintainers will get to it, and it can also lead to maintainer burnout. Projects heavy in rejected requests may benefit from providing more instructional information to their users to communicate expectations for change request type.

2.2.4 Technical Forks

Technical forks are instances of a code repository that an individual copies over to their platform account to use and modify. Forking is an excellent tool which allows contributors to experiment with project code without effecting the main source. Forks can be contributing forks, or forks which regularly open change requests, or non-contributing. The technical forks metric quantifies how many copies of a project are in distribution at any certain time. This metric is useful to track as it can indicate how users are interacting with the code using platform tools. A high number of forks for example, while not necessarily a project risk, may result in unnecessary difficulties in merging later if major conflicting contributions are made on different contributing forks without communication.

2.2.5 Code Change Lines

Like contribution information, quantifying the amount of actual physical code edits is a great way to provide a general volume of project activity. Code change lines is a metric which tracks the number of lines (added and removed) that are changed over a specific period. This metric provides granularity that contributions does not as the number of lines changed can be large or small within a single commit. Further filtration of this metric can provide information about authors of bulk changes, type of changes, and what the goal is of these changes (aggregating code lines for efficiency versus large blocks of code to add additional features).

3 System Use

The details of the systems intended actors, use cases, and requirements are described in detail below. This section will serve as a building block for the project and may be updated as further sprints are completed and information granularity increases.

3.1 Actors

The primary actors of this system are the Maintainers, Contributors, and Stakeholders.

Maintainer

The Maintainer is responsible for setting up the 8Knot web application with the appropriate databases and repositories and is responsible for configuring the data collection process using the software tools. The Maintainer can then use the data collected to reallocate resources, present

concise information to stakeholders / sponsors, recruit contributors, and generally communicate the overall health of the project. Maintainers are also responsible for the commit rights on the primary branch of the open-source project. The statistics provided through project health metrics can be useful in improving the experience of those that contribute to such projects, maintaining a wider database of contributors.

System Features:

- Install 8Knot software
- Add repositories from which data is to be collected
- Configure data collection methods
- Customize database dashboard using visualization tools as needed
- Dynamically interact (filter/sort) with database dashboard
- View database dashboard

Contributor

The Contributor is a user who has historically or is currently contributing to the open-source project which 8Knot is displaying data from. This user can review the relevant dashboards generated through the Augur instances to view meaningful contribution statistics, and whole picture project health, as determined by the Maintainer.

System Features:

- Dynamically interact (filter/sort) with database dashboard
- View data dashboard

Stakeholders / Company Sponsor

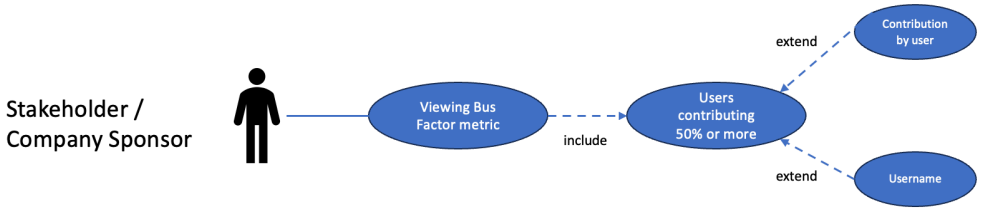
Stakeholders are those who do not necessarily contribute code / software to the project but are affected by the health and sustainability of the project in other ways. This can include those who invest financially in the project, businesses directly using the tool being updated, and other entities with external business interests in the project's health and sustainability.

System Features:

- Dynamically interact (filter/sort) with database dashboard
- View data dashboard

3.2 Use Case

Below represents a single identified use case for this project. More will be added as the project becomes more fully developed over the course of several sprints.



Title	Stakeholder Views Bus Factor Metric
Trigger	Action is initiated by Stakeholder navigating to Starter Health page on 8Knot dashboard site.
Primary Actor(s)	Stakeholder / Company Sponsor
Description	This use case describes how Stakeholders / Sponsors can interact with the 8Knot display dashboard to gain specific information about the number of contributors that make up the majority contributions for a project (bus factor)
Basic Flow of Events	Basic flow for the use case begins after the Stakeholder / Sponsor navigates their web browser to the 8Knot Dashboard – Starter Health page: 1. Contribution metric data in terms of the number of critical contributors is displayed near the top of the metrics model page. 2. Bus factor information can be adjusted to view for specific time periods and repositories of a project through dynamic fields associated with the visual. 3. As fields are adjust the pie-charts will be regenerated based on Stakeholder / Sponsor adjustments.
Alternative Flow of Events	If the flow of this use case is disrupted at any time: 1. The display will regenerate to its default and the web page will refresh. 2. The user can re-input any dynamic fields. 3. If problem persists webpage will present error message to Stakeholder describing what point of connectivity is causing the issue.
Special Requirements	Usability – The interfaces of the dashboard will be clearly labeled with how a user may dynamically interact with presented information. An “About Graph” button in the corner of each pie chart will give further detail about the construction of the graphs and how they can be read. Performance – The pie charts shall react promptly to changes within dynamic inputs, pending the user web connection.
Pre-Conditions	The Stakeholder / Sponsor must be on the correct website and navigated to the Starter Health dashboard page.

Post-Conditions	After the Stakeholder / Sponsor has finished examining domain name information, they may close the page.
Extension Points	N/A

3.3 Requirements

Below are the systems requirements that have been identified so far. They are divided into categories of general requirements which apply to the 8Knot modifications and system, and specific requirements for the Starter Health Metric and the Community Development Index Metric. As we iterate through further development of this effort these requirements are subject to change and will be expanded upon as necessary to create further definition.

3.3.1 General Requirements

Component	Priority	Requirement Name	Requirement Description
Database	High	Installation of 8Knot	Maintainer to be able to install 8Knot docker package, including all necessary software tools
Database	High	Configure 8Knot/Augur Repositories	Maintainer able link appropriate repositories for data collection.
Database	High	Control data collection start date and end date	Maintainer to configure what dates data is collected.
Database / Front End	Medium	Dashboard Display Web Pages	When User navigates to 8Knot dashboard, sites include Starter Health and Collaboration Development Index tabs'
Database / Front End	Medium	About Graph Button	Graphs to display push button option to pop-up window displaying summary of graph displayed information.
Database / Front End	Medium	Information	Web Information tab to include further details and definitions clarifying details about how metrics are collected, computed and what information they provide

User Authentication and Access Control	High	User Authentication	The system should implement user authentication to restrict access to authorized users.
Export and Reporting	Medium	Export to CSV	Allow users to export metrics data in CSV format for further analysis.
Manage User Access	High	User Permission Management	The system administrator should be able to manage user access and permissions.
Data Visualization	Medium	Dashboard Creation	The system must provide graphical representations and dashboards to visualize metrics data.
Documentation and Training	Low	User Documentation	Provide comprehensive user documentation explaining how to use the metrics system.

3.3.2 Starter Health Metric Model Requirements

Component	Priority	Requirement Name	Requirement Description
Bus Factor	Medium	Display type	Bus Factor metric to be displayed as a pie chart with contribution % by username, only those within the specified bus factor percent will be shown.
Bus Factor	Medium	Dynamic date selection for metric	Bus Factor metric to have dynamic start and end dates for which user can see contributors which represent bus factor for that time range

Bus Factor	Medium	Dynamic bus factor contribution amount selection.	Percentage of contribution used in bus factor determination (default to 50%) to be dynamic field able to be toggled up / down.
Time to First Response	Medium	Dynamic date selection for metric	Graph date range of response time metrics to be a dynamic field in terms of both start and end date, modifiable by metric viewer.
Time to First Response	Medium	Calculated and displayed moving average.	For the selected date range displayed on the time to first response graph a period average will be calculated and displayed.
Change Request Ratio	Medium	Dynamic date selection for metric	Graph date range of change request ratio to be a dynamic field in terms of both start and end date, modifiable by metric viewer.
Release Frequency	Medium	Dynamic date selection for metric	Graph date range of release frequency to be a dynamic field in terms of both start and end date, modifiable by metric viewer.

3.3.3 Community Development Index Model Requirements

Component	Priority	Requirement Name	Requirement Description
Data Collection and Storage	High	Data Retrieval	The system must collect, and store data related to code commits, contributor activity, and software downloads.
Metrics Calculation	High	Commit Frequency Calculation	The system should calculate the average number of commits per week in the past 90 days.
Generate Reports	Medium	Report Generation	Users should be able to generate reports based on metrics data for sharing with stakeholders.
Filter Metrics Data	Medium	Custom Filters	Provide the ability for users to apply custom filters to metrics data (e.g., by contributor name, repository).
Filter Metrics Data	Medium	Dynamic date selection for metric	Graph date for all graphs to be a dynamic field in terms of both start and end date, modifiable by metric viewer.

Filter Metrics Data	Medium	Trendline	Trendline to be displayed on graphs for code change commits and code change lines over selected date range.
Filter Metrics Data	Medium	Display of response type	Change request response type (accepted, rejected, with comments) to be displayed on graph for selected date range.

4 Interfaces

The interface of this system is for most actors a web-based client. The Maintainer will also interface with the system through database schema, and a command line interface which allows the Maintainer to modify the repositories being examined, control data collection methods, and tweak dashboard displays as necessary.

8Knot runs on an instance of Augur to collect open-source project data and provide this information to users. 8Knot runs utilizing a docker container, and a python frontend interface. The Augur credentials used to pull data for the system are passed as a .list file. Augur itself uses several technologies to function. The raw data for analysis comes from GitHub commit logs, GitHub's API, The Linux Foundation's Core Infrastructure Initiative API, and Succinct Code Counting (Augur Documentation, 2023). Augur utilizes Python (v3.08 to v3.10) and nginx to deploy. PostgreSQL is used by Augur to store data collected. Augur can be configured with Ubuntu 22.x as well as OSX.

5 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/