SPRINT 1: REQUIREMENTS

The goal of our metric visualizations is to give a project manager a tool for measuring how active the contributors to their project are. The visualizations will give insight into the specific areas where improvements could be made, like checking to make sure people are reviewing code, being more than one-time users, and communicating.
This tool is intended to prevent projects from falling into poor health and keep contributors engaged with the community.
To achieve this, 9 metrics are pulled from Chaoss:
STARTER PROJECT HEALTH
1) time to first response
2) change request closure ratio
3) bus factor
4) release frequency
COMMUNITY ACTIVITY
1) commit_frequency
2) meeting_count
3) meeting_attendee_count
4) code_review_count
5) contributor_count

Interested actors in the Starter Project Health metric model may include maintainers, individual contributors, project evangelists, and anyone else who plays a part in the active development of the project.
Interested actors in the Community Activity metric model would likely include 3rd parties who are interested in utilizing the project, as it would allow them to tell whether they can rely on the project to be continually developed. It may also be useful to sponsors who are trying to determine where they should place their support for open source projects.

These metrics are used to build these visualizations:
STARTER PROJECT HEALTH
1) time to first response - response time is crucial to keeping contributors engaged.
2) bus factor - the bus factor should be checked to make sure the community stays healthy.
3) release frequency - this should be checked to make sure the community is active.
4) change request closure ratio - this should be monitored and kept relatively even.
COMMUNITY ACTIVITY
1) communication - contributor count / average meeting attendee count, important for measuring how communicative the contributors are.
2) meeting count - important for measuring how frequently the team comes together to discuss implementation.
3) commit frequency - this should be checked to ensure the community is active in a similar manner to release frequency.
4) code review count - important for ensuring that contributors are ensuring quality.

System Design

1.Introduction

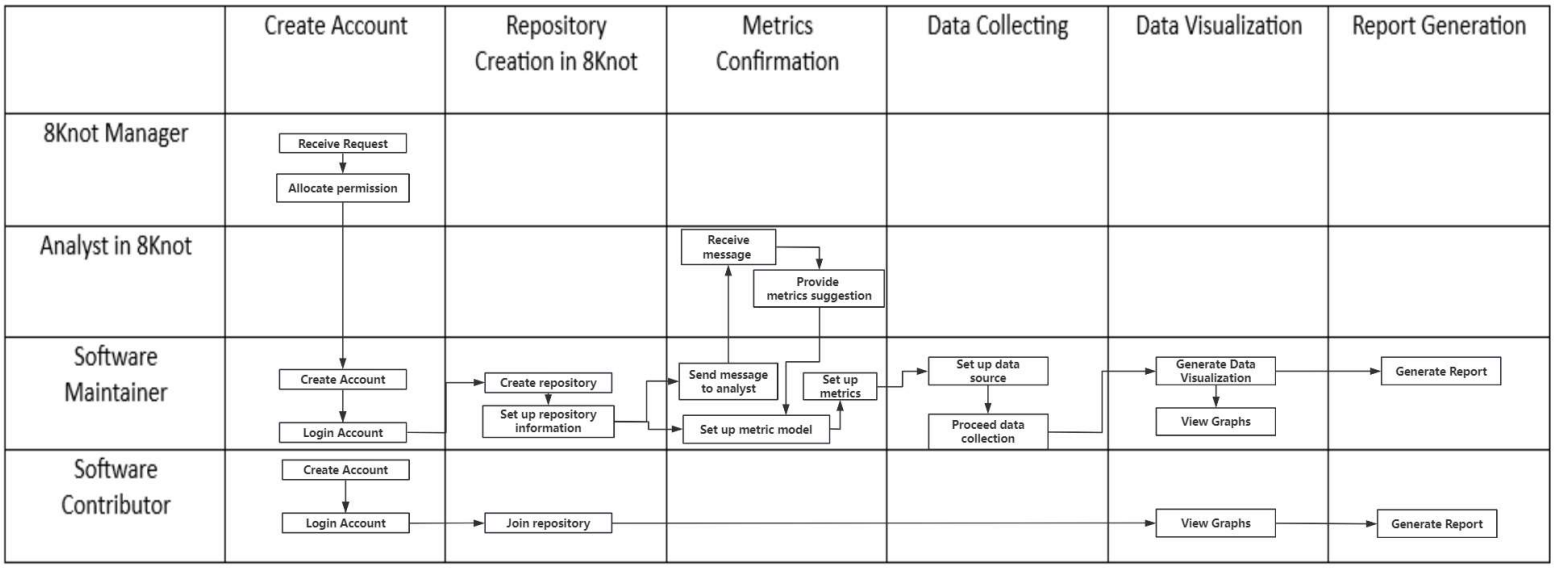
The system aims at providing metrics to supervise the health and sustainability of software, which helps

the software maintainers have better work on making improvement.

The web applications contain all features needed by repositories maintainers. It includes authentication,

repository management, collecting data, data storage, communicating between analyst and software maintainers, generate reports.

2.Workflow



3.Requirements

User	Activity	Associated Data	Description
8Knot Manager	Receive Request	maintainerUserID	8Knot manager should be able to receive request for access of the metric model system
8Knot Manager	Allocate Permission	maintainerUserID	8Knot manager should be able to receive request for access of the metric model system
Analyst in 8Knot	Receive Message	MessageID;repositoryID	Analyst in 8Knot should be able to receive message from the Software Maintainer who asking for suggestion on metric models
Analyst in 8Knot	Provide Metrics Suggestion	MessageID;repositoryID	Analyst in 8Knot should be able to provide metrics suggestion for the Software Maintainer who asking for suggestion on metric models
Software Maintainer	Create Account	maintainerUserID	Software Maintainer should be able to create account
Software Maintainer	Login Account	maintainerUserID	Software Maintainer should be able to login account
Software Maintainer	Create Repository	repositoryID	Software Maintainer should be able to create repository
Software Maintainer	Set Up Repository Information	repositoryID;repositoryContent	Software Maintainer should be able os set up repository information for the repository he/she creates

Software Maintainer	Send Message to Analyst	MessageID;repositoryID	Software Maintainer shoud be able to send message to the Analyst in 8Knot for suggestion on metric models
Software Maintainer	Set Up Metric Model	repositoryID;MetricModelID	Software Maintainer should be able to set up metric models for the repository he/she creates
Software Maintainer	Set Up Metrics	MetricModelID;MetricID	Software Maintainer should be able to set up metrics in on specific metric model
Software Maintainer	Set Up Data Source	repositoryID;MetricModelID	Software Maintainer should be able to set up data sources for the repository he/she creates
Software Maintainer	Proceed Data Collection	repositoryID;MetricModelID;MetricID;DataID	Software Maintainer should be able to start the data collection process
Software Maintainer	Generate Data Visualization	repositoryID;MetricModelID;MetricID;DataID	Software Maintainer should be able to get data visualization using the data collected and the metric models
Software Maintainer	View Graphs	repositoryID;MetricModelID;MetricID;DataID	Software Maintainer should be able to view the result of the data visualization
Software Maintainer	Generate Report	repositoryID;MetricModelID;MetricID;DataID	Software Maintainer should start the generate report from the data visualization
Software Contributor	Create Account	contributorUserID	Software Contributor should be able to create account
Software Contributor	Login Account	contributorUserID	Software Contributor should be able to login account
Software Contributor	Join repository	repositoryID	Software Contributor should be able to join repository
Software Contributor	View Graphs	repositoryID;MetricModelID;MetricID;DataID	Software Contributor should be able to view the result of the data visualization
Software Contributor	Generate Report	repositoryID;MetricModelID;MetricID;DataID	Software Contributor should be able to generate report from the data visualization

4.Use Case

Use-case name	Software maintainer Send Message to Analyst
System or subsystem	8Knot
Actors	Software maintainer
Brief Description	The software maintainers send message to the analyst for suggestion on choosing data model.
Basic flow of events	The basic flow begins after the repository log in the account, create a repository and click into the repository page. When sending message to the analyst, the analyst can receive message from the corresponding repository information page.

	1. The repository page shows a click button, which to open a chat dialog. 2. The software maintainer input the project information in the input field 3. After clicking the “send” button, the message will be sent to the web server, and a dialog will pop up to show the message is sent successfully.
Alternative flow of events	Sending message attempt failed: 1. The repository maintainer did not click “send” button and close the chatting dialog or the window. No message would be sent and recorded 2. The internet is not connected. A dialog would pop up to notify user. 3. The web server did not respond to the request. A dialog would pop up and remind the user to connect website owner for solution.
Special requirements	Performance: Sending message response time has to be no more than 5 seconds after the use click “send” button.
Pre-conditions	The software maintainer has log in the account. The software maintainer has created a repository. The software maintainer has been on the repository page.
Post-conditions	A message notification would show up in analyst’s account page. The message notification will lead the analyst to corresponding repository page. The message sent would show in the chatting dialog of corresponding repository page in the software maintainer‘s account.
Extension points	