Putting Order into CHAOSS

Metrics to Analyze Code Development

Ana Jiménez Santamaría, Daniel Izquierdo Cortázar
@bitergia, https://speakerdeck.com/bitergia

CHA OSSCon Europe - Brussels - 1st Feb. 2019
About us:

Ana Jimenez
@anajsana95
Marketing at bitergia, software development analytics Gamer and anime lover Self-taught illustrator 日本語の学生 España - USA

Daniel Izquierdo
@dizquierdo
@Bitergia co-founder. Open and Inner Source advocate. Love data analytics. Python developer.
Motivation

- Understand what CHA OSS-GMD is
- Contribute (& learn how to do it)
- Put metrics into action [integrate soft.&metrics]
- New panels!
C.H.A.O.S.S = Community Health Analytics Open Source Software

CHAOSS Growth, maturity and decline group
Software development projects life cycle:

Growth, maturity and decline
How can I know my community health?

Identifying my project state: Growth Maturity or Decline?

- Code development
- Community growth
- Issue resolution
Contributing

You can start by introducing yourself on the CHA OSS mailing list (see below) explaining your interest. Then, you can have a look at the archives of the mailing lists, at the minutes of past meetings, and at the

- issues and
- pull requests in
- contents of this repository.

To contribute, of course you can participate in the in online meetings, and in issues and pull requests. Currently, main lines of work are:

- Use cases. Proposal and discussion of use cases that help to understand metrics in context. You can propose your use cases, and/or contribute to refine those that have been proposed already.

- Focus areas (definitions, goals, questions). Currently, in the [definition of GMD metrics] we strucutre it in the following focus areas: Issue Resolution, Code Development, and Community Growth.

For each area of interest, we're following the goal-question-metric methodology defining questions and metrics that help to answer them.

You can contribute by proposing new goals for a focus area, or new questions for learning about those goals, or new metrics for answering those questions. Or by helping to refine goals, and questions.

See more details in the contributing file.
We're in the process of discussing questions and metrics. Meanwhile, below is the former list of metrics, and their related questions.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pull Requests Merged</td>
<td>What is the number of code commits?</td>
</tr>
<tr>
<td>Lines of Code Changed</td>
<td>What is the number of lines of code changed?</td>
</tr>
<tr>
<td>Code Reviews</td>
<td>What is the number of code reviews?</td>
</tr>
<tr>
<td>Pull Request Merge Duration</td>
<td>What is the duration of time between code merge request and code commit?</td>
</tr>
<tr>
<td>Code Review Efficiency</td>
<td>What is the number of merged code changes/number of abandoned code change requests?</td>
</tr>
<tr>
<td>Maintainer Response to Merge Request Duration</td>
<td>What is the duration of time for a maintainer to make a first response to a code merge request?</td>
</tr>
<tr>
<td>Code Review Iteration</td>
<td>What is the number of iterations that occur before a merge request is accepted or declined?</td>
</tr>
<tr>
<td>Forks</td>
<td>Forks are a concept in distributed version control systems like GitHub. It is a proxy for the approximate number of developers who have taken a shot at building and deploying the codebase for development.</td>
</tr>
<tr>
<td>Pull Requests Open</td>
<td>Number of open pull requests.</td>
</tr>
<tr>
<td>Pull Requests Closed</td>
<td>Number of closed pull requests.</td>
</tr>
<tr>
<td>Pull Request Comment Duration</td>
<td>The difference between the timestamp of the pull request creation date and the most recent comment on the pull request.</td>
</tr>
<tr>
<td>Pull Request Comment Diversity</td>
<td>Number of each people discussing each pull request.</td>
</tr>
<tr>
<td>Pull Request Comments</td>
<td>Number of comments on each pull request.</td>
</tr>
</tbody>
</table>
Contributing
[for a newcomer]
Contributing

Open Ticket & Discussion

Open ticket at grimoirelab-sigils
Have discussion if needed

Creation Process

Panel
Documentation
Export panel as JSON file
Export documentation as md file.
Pull Request to grimoirelab-sigils

Review & Merge

The maintainer may look for improvements
(There were indeed!)
When ready to go, merge the panel.
Putting metrics into action:

Lines of code changed panel visualization:

This panel focuses on the **Number of lines of code changed**. We based the panel on Git data.

The idea behind the panel is having a look on how community contributes to the repositories by having a look at the magnitude of those changes with several metrics related with Git commits, specially modified lines and the amount of affected files. By using this panel, we can identify and analyze behaviors of specific organizations, projects or repositories.

There are some filters added by default:

- **Add Merge**: It ignores those commits whose number of modified files is 0 (this is, commits from merges).
- **Add Date**: It ignores those commits whose author is marked as a Bot.

Here is the link to the full documentation for this panel.
Putting metrics into action:

Pull Request Merge Duration panel visualization:

Dashboard / Pull Request Merge Duration
Search... (e.g. status:open AND extension:PHP)

Pull Requests Add a filter

Big numbers

2,108 total pull request
0.7 median time to close (days)

Time to Close days

median time to close (days)

Repos: 20

This panel focuses on pull request merge duration, defined by the time between a code merge request and code commit. In this case, we focus on GitHub pull requests.

When measuring GitHub, we cannot identify if a pull request has been merged, only if it is open or closed. That’s the reason why we will base our panel on pull request closing time.

The main idea behind this panel is to help us identify a project state (growth maturity or decline) by answering custom queries like:

- How long does a code merge request take in order to be committed?
- What organizations/authors are being more efficient when closing a pull request?
- Is my project taking too long when closing code merge requests?

Here is the link to the full documentation for this panel.

Bitergia
Putting metrics into action:

Pull Requests Merged

This panel focuses on **merged** GitHub Pull Requests. As GitHub API does not always identifies merges, we based the panel directly on Git data.

The idea behind the panel is having a look on how community performs and evolved merging Pull Requests. By using this panel we can identify trends and analyze behaviours of specific organizations, projects or repositories.

**Commits**: a filter is applied within visualizations to get only those that we consider as "merges". We consider a commit as a "merge" if satisfies at least one of the following conditions:

- Number of files is 0.
- Commit message starts with "Merge"
Maintainer First Response

Dashboard / Maintainer Response to Merge Request Duration

Search... (e.g., status:200 AND extension:PHP)

Add a filter

Filter

Organization
Selected...

Project
Selected...

Apply Cancel changes Clear form

Summary

<table>
<thead>
<tr>
<th></th>
<th>0.03</th>
<th>2.394</th>
<th>2.330</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median Time to First Response (Days)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average Time to First Response (Days)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td># Pull Requests</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Repository Data

<table>
<thead>
<tr>
<th>Repository</th>
<th>Average Time to First Attention (Days)</th>
<th>Median Time to First Response (Days)</th>
<th># Pull Requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>chaos/mirrornlab-tutorial</td>
<td>19.824</td>
<td>0.42</td>
<td>42</td>
</tr>
<tr>
<td>chaos/mirrornlab-sortinghat</td>
<td>12.596</td>
<td>0.485</td>
<td>101</td>
</tr>
<tr>
<td>chaos/mirrornlab</td>
<td>7.333</td>
<td>1.595</td>
<td>101</td>
</tr>
<tr>
<td>chaos/mirrornlab-hatul</td>
<td>5.517</td>
<td>0.29</td>
<td>248</td>
</tr>
<tr>
<td>chaos/mirrornlab-summer</td>
<td>7.13</td>
<td>0.045</td>
<td>201</td>
</tr>
<tr>
<td>chaos/mirrornlab-summer-summer</td>
<td>6.64</td>
<td>0.715</td>
<td>51</td>
</tr>
<tr>
<td>chaos/mirrornlab-metrics</td>
<td>4.239</td>
<td>0.46</td>
<td>82</td>
</tr>
<tr>
<td>DDSHealth/health</td>
<td>3.584</td>
<td>0.4</td>
<td>67</td>
</tr>
<tr>
<td>chaos/mirrornlab-diversity-inclusion</td>
<td>2.84</td>
<td>0.67</td>
<td>93</td>
</tr>
<tr>
<td>chaos/mirrornlab-manuscripts</td>
<td>2.824</td>
<td>0.02</td>
<td>85</td>
</tr>
</tbody>
</table>

Help

This panel focuses on any GitHub Pull Request and measures its first time to attention.

The idea behind the panel is having a look on how community performs and evolves responding to pull requests. By using this panel we can identify trends and analyze certain behaviours of specific organizations, projects or repositories.

- **Summary**: This widget contains a summary of the information about the time to first response in average and in median.
- **Repository Data**: This table provides an overview of the same information but split by repository in GitHub.
- **Average Time to First Response in Days over Time**: This evolutionary chart displays the behaviour of the average metric over time.

More in-depth documentation can be found at the Maintainer Merge Request Duration Page
<table>
<thead>
<tr>
<th>Conclusions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>What we achieved</strong></td>
</tr>
<tr>
<td>Understand</td>
</tr>
<tr>
<td>Contribute</td>
</tr>
<tr>
<td>Put metrics into action</td>
</tr>
<tr>
<td>New panels!</td>
</tr>
</tbody>
</table>
Let's go for questions!

Ana Jiménez, Daniel Izquierdo

anajs|dizquierdo@bitergia.com