Mining Development Data to Understand and Improve Software Engineering Processes in HPC Projects
(the slides are available under “Presentation Materials” in the above URL)
Date: July 7, 2021
Presented by: Boyana Norris (University of Oregon)

Q. Does the size or maturity of a project affect the way that your analysis methods work?

A. No, the techniques are the same regardless of the amount of data available; one can choose to restrict the time period on which to focus, but in general the metrics can be computed on any kind of project. There are some projects that are lacking certain types of data or enough information (e.g., brand new ones) to look at trends.

Q. Re: the LOCC metric: Some commits make global search-and-replace type changes (for example renaming an important internal class, or applying a new whitespace convention), that might touch a large fraction of the codebase (high LOCC, many lines in many files). However such changes are not well-correlated with developer effort or cognition. Do you try and recognize or filter this type of activity during analysis?

A. The LOCC metric is indeed insufficient; hence, all change-related analyses are parameterized by the change metric, allowing the use of any text-difference based methods available. We use the textdistance Python package which provides ~30 methods for computing similarity between two text. For example, the cos distance after a global reformatting would be close to 0, while LOCC would be very large.

Q. (This is a slightly different take on the LOCC question above) I’m wondering if there might be ways to distinguish between productive changes (such as blocks of code that are added and remain relatively stable) vs. potentially unproductive changes (blocks that are constantly being modified by both additions and deletions).

A. Yes, I think that is a very interesting idea. We do not yet classify the changes based on such categories, but that’s a logical extension. You can accomplish this by defining a new metric that tracks the frequency of changes of all modified chunks of code, possibly weighted by how recent the changes were.

Q. How do Github/Gitlab web dashboards help HPC-specific metrics such as productivity versus developing web apps?

A. I think they are a great resource for seeing “at a glance” activity in a project, in terms of simple counts (commits, developers, etc.), but without any ability to define custom metrics, the utility is limited. While we don’t provide a dashboard (yet), we aim to enable extensible and easy to code analyses; I think these are complementary capabilities. Contributor CI looks like very cool tool with interesting visualizations; I’d be very interested in exploring connections.

Q. And does "in the zone" only include lines of code merged into main branches? There are likely many PRs that don't get addressed where people spend a significant amount of time - people can be burning out without showing up on that chart.

A. In the zone can be defined over all the code or some subset of the branches. We are in fact going to include (in August) an example that shows abandoned PRs, based on your question -- thank you!

Q. Do you try and distinguish between changes to code files and in-repo documentation (including comments, READMEs, markdown files, etc)?

A. Yes, for all files with established suffixes (e.g., *.md, *.html, etc), which is done automatically. However, projects sometimes have custom suffix rules, and in that case, the process still relies on some manual labeling which is not ideal.

Q. Are your tools available in a containerized environment?

A. They are now, thank you for contributing!

Q. With the changes in Python versions and other tools. Maybe?

A. I am not sure if I understand this question, but if it’s in regards to containers, then this has now been contributed and merged into the main branch.