The Comment Density of Open Source Software Code

Oliver Arafat[i]
Siemens AG, Corporate Technology
Otto-Hahn-Ring 6, 81739 Munich, Germany
oarafat@gmail.com

Dirk Riehle
SAP Research, SAP Labs LLC
3412 Hillview Ave, Palo Alto, CA 94304, USA
dirk@riehle.org

Abstract

The development processes of open source software are different from traditional closed source development processes. Still, open source software is frequently of high quality. Thus, we are investigating how open source software creates high quality and whether it can maintain this quality for ever larger project sizes. In this paper, we look at one particular quality indicator, the density of comments in open source software code. In a large-scale study of more than 5,000 projects, we find that active open source projects document their source code, and we find that the comment density is independent of team and project size, but not of project age. In future work, we intend to correlate comment density with project success or failure.

1. Introduction

Open source software has become an important part of commercial software development and use [1]. Most interestingly, open source projects have reached a size and complexity that rivals the size of some of the largest commercial projects [2], yet they are being developed in a manner quite different from traditional software engineering processes.

Our research goal is to improve our understanding of open source software development processes and to transfer appropriate practices into corporate software development. This has become particularly important, because the traditional life-cycle model or the more recent agile methods either don't scale to large project sizes or have problems in coping with changing requirements.

In this study we focus on one particular code metric, the comment density, and assess it across 5,229 active open source projects, representing about 30% of all active open source projects. Comment density is the percentage of comment lines in a given source code base, that is, comment lines divided by total lines of code. Comment density is assumed to be a good predictor of maintainability and hence survival of a software project [3] [12] [13].

The contributions of this study are the following: For the first time, we assess the comment density of open source on a large scale, demonstrate that commenting is an integral practice of open source software development, and show that the comment density of active open source projects is independent of team and project size but not of project age.

The paper is organized as follows. Section 2 discusses related work, Section 3 discusses our approach, Section 4 presents our results, and Section 5 discusses future work and some conclusions.

2. Related Work

Prechelt reports about a controlled experiment performed from 1997-1999 [11]. Prechelt found that scripting language solutions were significantly better documented than non-scripting language solutions. Values for the comment densities were in the 20-30% range. Prechelt's subjects were students, and the programs were throw-away exercises.

Sundbakken assess the comment density of maintenance phase code contributions to components of four open source projects [4]. Sundbakken observes that consistent commenting correlates highly with maintainability of components. The measured comment density ranges from 0.09% for poorly maintainable components to 1.22% for highly maintainable components.

In contrast to Sundbakken, in a study on the comment density of a closed-source compiler project in its maintenance phase, Siy and Votta find a consistent comment density of around 50% [5].

In another study of 100 Java open source classes, Elish and Offutt find an average comment density of 15.2% with a standard deviation of 12.2% [7].

Fluri et al. present an approach for assessing the comment density of software projects and demonstrate the approach using three selected open source projects [13]. Comment densities for the exemplary projects vary widely. They also observe that new code is barely commented, implying that the comment density decreases over time.

Among other things, our work improves over the state of the art by being the first large-scale study that goes beyond a few selected case studies.

3. Approach

We use the database of the open source analytics firm Ohloh, Inc. [8]. We work with a database snapshot of March 2008, but have cut off all analysis data after December 31st, 2007. The database contains detailed data from about 10,000 open source projects.

We are only interested in active well-working open source projects, not dead projects. We define and apply an active project filter to let a project pass only if by the end of 2007 it was at least two years old and

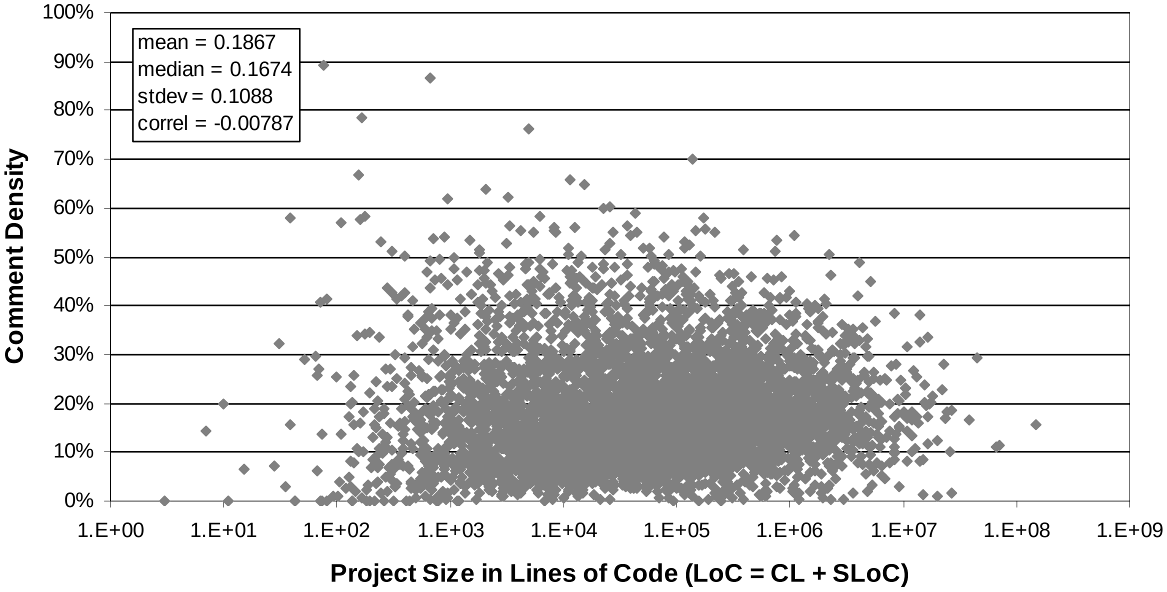

Figure 1: Comment density as a function of lines of code for a given project.

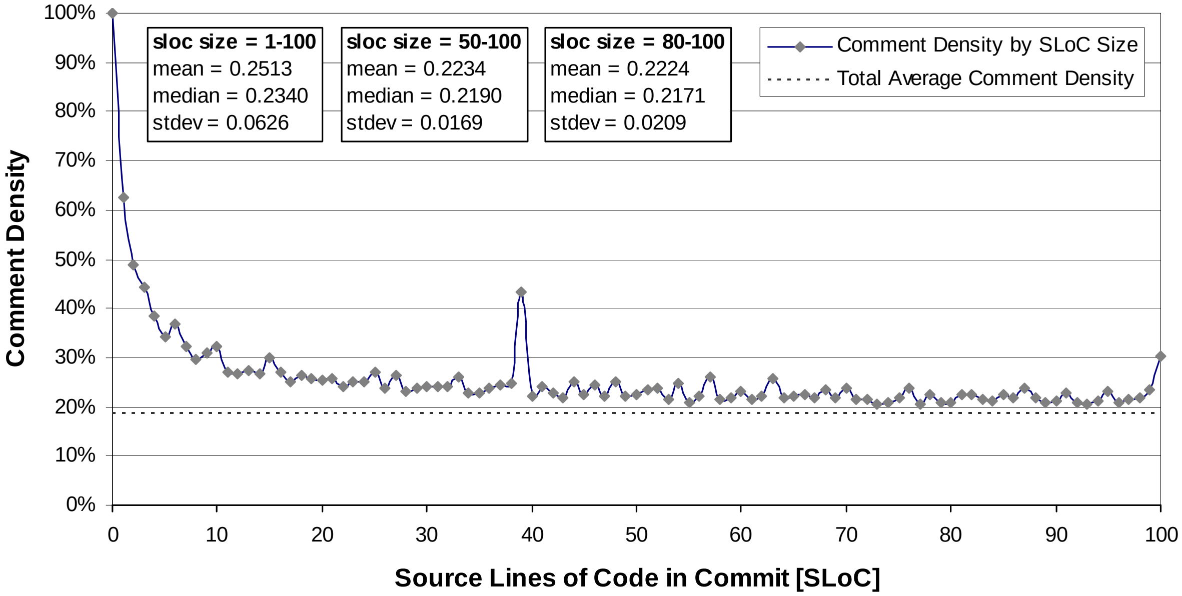

Figure 2: Comment density as a function of source code lines in a given commit.

if the code activity of the last year had been at least 60% of the activity of the previous year. This filter reduces the original 10,000 projects to 5,229 projects. Using a comparable approach, Daffara estimates that there were about 18,000 active open source projects in the world by August 2007 [6], so our sample represents about 30% of the total population.

The code contribution history of a project is a time series of commits (code contributions) to the code repository. A ***commit*** represents a set of changes to the source code performed as one chunk of work. We apply filters to improve data quality. For example, we filter out file rename and move operations.

- A ***source line of code***, or SLoC, is a physical line in a source file that contains source code.
- A ***comment line***, or CL, is a physical line in a source file that represents a comment.
- A ***line of code***, or LoC, is either a source line of code or a comment line.

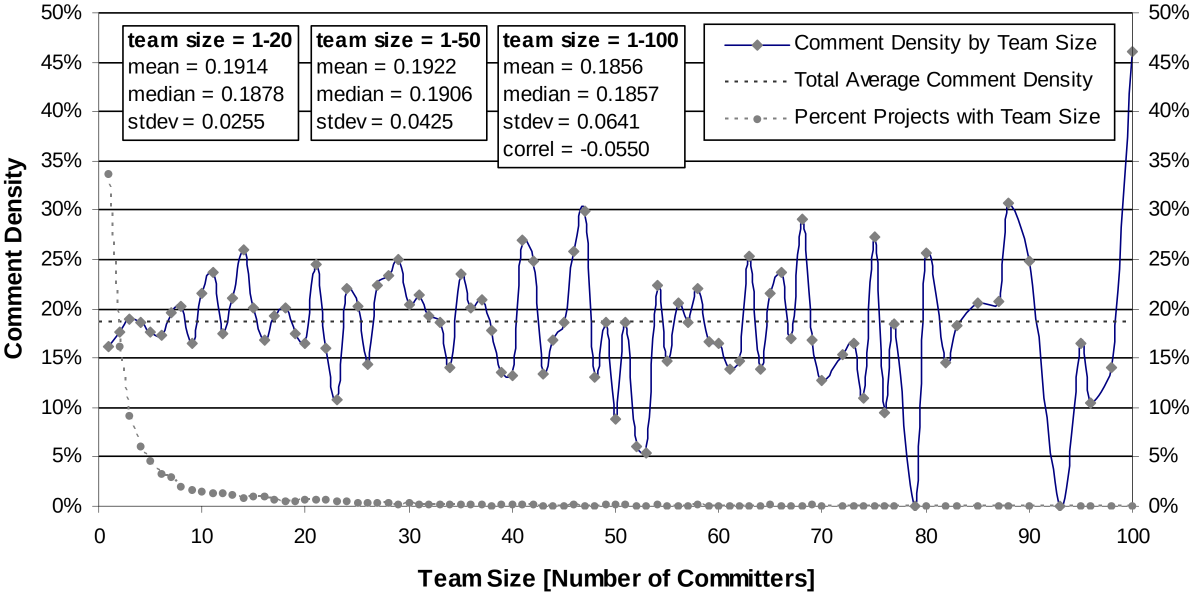

Figure 3: Comment density as a function of team size of open source projects.

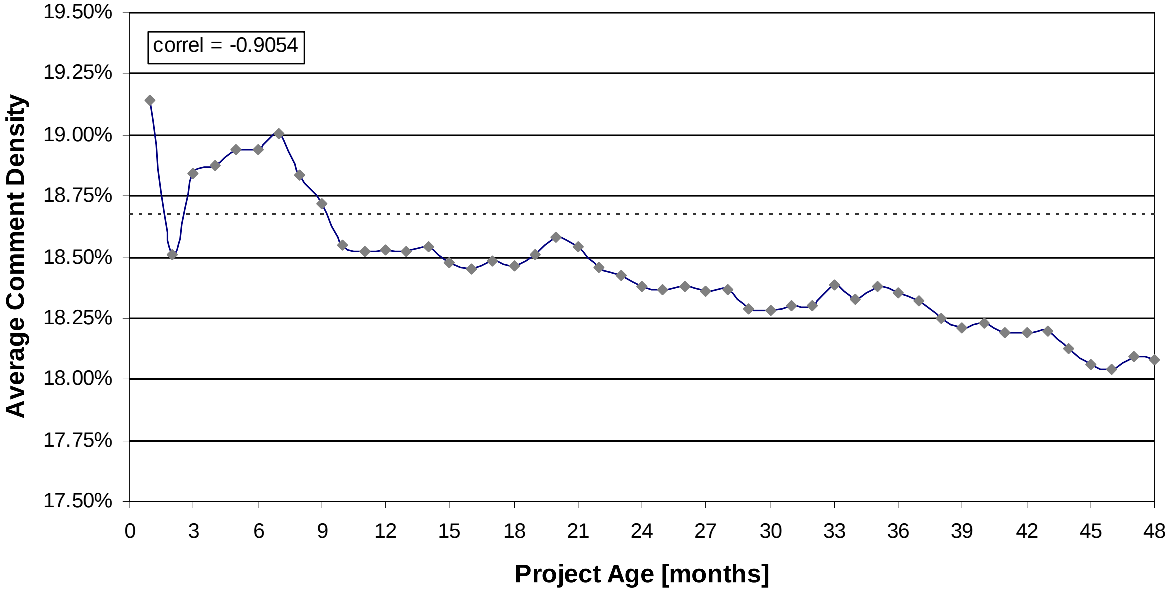

Figure 4: Comment density as a function of the age of open source projects.

The ***commit size*** of a commit is the number of lines of code affected by a commit, whether added, removed, or changed [10].

The ***comment density*** of a file or a group of files or the whole source code base of a project is defined as the number of comment lines divided by the number of lines of code of the same code body [3].

We use a tool chain that consists of the original database in a PostgreSQL RDBMS instance, intermediate processing using SQL queries and Java code, and final processing using the R project and Excel.

4. Results and Limitations

We extracted the data visualized in Figures 1 to 4. Figure 1 shows the comment density in our sample population. The average value is 19% so about 1 line of code in 5 lines is a comment line. In our population the comment density varies significantly. Figure 2 shows that on average, small commits have a better than average comment density, suggesting that commenting is an integral part of programming.

Figure 1 also shows a correlation of -0.0079 between project size and comment density; thus project size and comment density are independent of each other. Figure 3 finds a correlation of 0.0255 between team size and comment density for the majority of projects (team size < 20 committer). Thus, team size and comment density are also independent of each other. Figure 4 shows a correlation of -0.9054 between project age and comment density, however, the actual decrease in overall comment density after four years (48 months) is rather small.

An important limitation of this study is that we consider all comment lines as equal, whether they provide rich content or are auto-generated stubs. (The diff tool/parser distinguishes programming languages and recognizes multi-line comments though [9].) We don't discuss the impact of programming languages on comment density due to the lack of space.

We analyze only active projects and have yet to determine to what extent a high comment density can be used as a predictor of project success or failure.

5. Conclusions

We have found that commenting source code is a consistent practice of active open source projects. It has led to an average comment density of about 19%. This density is maintained by dedicated commenting activities (about 2.5% of all code contributions) as well as in regular on-going programming activities.

Also, we have found that the average comment density is independent of team size and project size, suggesting that as teams and projects get larger, successful open source projects maintain their commenting discipline. However, the average comment density is not independent of a project's age but rather declines with an aging project. That decline is statistically significant; however, it is rather small and thus has limited practical implications.

6. References

[1] Amit Deshpande, Dirk Riehle. "The Total Growth of Open Source." In *Proceedings of Fourth Conference on Open Source Systems.* Springer, 2008. Page 197-209.

[2] Diomidis Spinellis. "A Tale of Four Kernels." In *Proceedings of the 2008 International Conference on Software Engineering* (ICSE '08). IEEE Press, 2008. Page 381-390.

[3] N. E. Fenton. *Software Metrics: A Rigorous and Practical Approach.* Thomson Computer Press, 1996.

[4] Marius Sundbakken. *Assessing the Maintainability of C++ Source Code.* M.S. Thesis, Washington State University, 2001.

[5] Harvey Siy, Lawrence Votta. "Does the Modern Code Inspection have Value?" In *Proceedings of the 17th IEEE International Conference on Software Maintenance* (ICSM '01). IEEE Press, 2001. Page 281-290.

[6] Carlo Daffara. "How Many Stable and Active Libre Software Projects?" See http://flossmetrics.org/news/11.

[7] Mahmoud Elish, Jeff Offutt. "The Adherence of Open Source Java Programmers to Standard Coding Practices." In *Proceedings of the 6th IASTED International Conference Software Engineering and Applications*. Page 193-198.

[8] Ohloh, Inc. See http://www.ohloh.net.

[9] Ohloh, Inc. ohcount. See http://labs.ohloh.net/ohcount.

[10] Philipp Hofmann, Dirk Riehle. "Estimating Commit Sizes Efficiently." In *Proceedings of OSS '09*, forthcoming.

[11] Lutz Prechelt. "An empirical comparison of C, C++, Java, Perl, Python, Rexx, and Tcl for a search/string-processing program." *Technical Report 2000-5*, Universität Karlsruhe, Fakultät für Informatik, Germany, March 2000.

[12] David Parnas. "Software Aging." In *Proceedings of the 16th International Conference on Software Engineering* (ICSE 1994). Page 279-287.

[13] Beat Fluri, Michael Wursch, and Harall Gall. "Do Code and Comments Co-Evolve? On the Relation Between Source Code and Comment Changes." In *Proceedings of the 14th Working Conference on Reverse Engineering* (WCRE 2007). Page 70-79.

[i] Work performed while working at SAP Research, SAP Labs LLC.