Chapter 10
Measuring Open Source Quality: A Literature Review

Claudia Ruiz
Georgia State University, USA

William Robinson
Georgia State University, USA

ABSTRACT

There is an ample debate over the quality of Free/Libre Open Source Software (FLOSS) with mixed research results. The authors show that a reason for these mixed results is that quality is being defined, measured, and evaluated differently. They report the most popular approaches including software structure measures, process measures, and maturity assessment models. The way researchers have built their samples has also contributed to the mixed results with different project properties being considered and ignored. Because FLOSS projects evolve with each release, their quality does too, and it must be measured using metrics that take into account their communities' commitment to quality rather than just the structure of the resulting code. Challenges exist in defining what constitutes a defect or bug, and the role of modularity in affecting FLOSS quality. The authors suggest three considerations for future research on FLOSS quality models: (1) defect resolution rate, (2) kind of software product, and (3) modularity—both technical and organizational.

INTRODUCTION

Crowston et al. (2012) pointed out that studies that compared the quality of FLOSS with proprietary software showed mixed results. They suggested that these results vary greatly by project and proposed further research on the antecedents of quality (Crowston, Wei, Howison, & Wiggins, 2012).

DOI: 10.4018/978-1-4666-2937-0.ch010

This paper takes a first step towards addressing this issue by reviewing the FLOSS literature in order to answer the questions of what is quality and how is it measured. If different studies evaluate quality using different measures it will be like comparing apples and oranges. This would explain the mixed results of the FLOSS quality studies.

Because of quality's extreme subjectivity, it is not surprising that studies comparing the quality of FLOSS with proprietary in-house developed

software have produced mixed results (Kuan, 2003; Paulson, Succi, & Eberlein, 2004; Raghunathan, Prasad, Mishra, & Chang, 2005; Stamelos, Angelis, Oikonomou, & Bleris, 2002). The two main explanations for these results are that (1) each study has defined and measured quality differently and that (2) each study has evaluated different characteristics of FLOSS projects.

Defining quality differently will of course produce mixed results, but even when studies define quality in similar terms, they evaluate it using dissimilar criteria to select sample projects and project characteristics.

In order to understand what it is about certain FLOSS projects that lead them to produce high quality software, the antecedents of FLOSS quality must be found, and the first step to finding those antecedents is to agree on a definition and measure of quality.

The rest of the paper is organized as follows: the next two sections provide a brief background on FLOSS and Software Quality; then, we present our literature review Methodology followed by Findings and a Discussion of the implications of the findings.

FLOSS

A FLOSS project is one that offers its software under a license that is in accordance with the criteria in the Open Source Definition (OSI, 2006) providing for free redistribution of the compiled software and the openly accessible source code.

Linux, Apache, Firefox are commonly found in many computers today and were developed under open source licenses. Apache is a Web server used by 60% of Websites worldwide (von Hippel & von Krogh, 2003) and 23.2% of European and 14.5% of North American Web surfers use the Firefox Web browser (Hales, 2006).

This growing popularity begs the question: is FLOSS "better" than proprietary in-house developed software? Proprietary in-house developed software projects are considered successful if they finish on time, on budget, and meet specifications. But the same standards cannot be applied to judge the success of FLOSS projects, because they usually have minimal budgets, are always in a state of development, do not have an official end time, and do not have formal specifications (Scacchi, 2009).

This lack of objective measures of success has not deterred the adoption of FLOSS products. It even has become a common assumption that FLOSS products are of higher quality than traditionally developed software (Ajila & Wu, 2007; Stewart & Gosain, 2006) with firms entering FLOSS projects citing FLOSS's "quality and reliability" as one of the main motivating reasons for the endeavor (Bonaccorsi & Rossi, 2006).

This assumption can be traced back to Linus's law, which says that "given enough eyeballs, all bugs are shallow" (Raymond, 1999). This means that FLOSS's public peer review and frequent releases lead to fewer bugs because there are more people looking at the software, reporting errors, and fixing those errors. This assumption has a kernel of truth: it has been observed in the Apache project that most problem (bug) reports and solutions in FLOSS projects are contributed by periphery community members and less so by core developers (Rigby, German, & Storey, 2008).

Mature FLOSS projects are composed of a community, whose structure has been described as being like an "onion" with the most actively contributing members, who are the most invested in the project and have the greatest decision power in the inner part and the least contributing members with the least amount of decision power on the outside. The project leader is at the center and radiating out are the core members, the active developers, the peripheral developers, the bug fixers, the bug reporters, the readers, and the passive users (Ye & Kishida, 2003). These roles are dynamic, changing as the community evolves as the system they are building evolves (Ye & Kishida, 2003).