Chapter 4

OSS-TMM: Guidelines for Improving the Testing Process of Open Source Software

Sandro Morasca
Università degli Studi dell'Insubria, Italy

Davide Taibi
Università degli Studi dell'Insubria, Italy

Davide Tosi
Università degli Studi dell'Insubria, Italy

ABSTRACT

Open Source Software (OSS) products do not usually follow traditional software engineering development paradigms. Specifically, testing activities in OSS development may be quite different from those carried out in Closed Source Software (CSS) development. As testing and verification require a good deal of resources in OSS, it is necessary to have ways to assess and improve OSS testing processes. This paper provides a set of testing guidelines and issues that OSS developers can use to decide which testing techniques make most sense for their OSS products. This paper 1) provides a checklist that helps OSS developers identify the most useful testing techniques according to the main characteristics of their products, and 2) outlines a proposal for a method that helps assess the maturity of OSS testing processes. The method is a proposal of a Maturity Model for testing processes (called OSS-TMM). To show its usefulness, the authors apply the method to seven real-life projects. Specifically, the authors apply the method to BusyBox, Apache Httpd, and Eclipse Test & Performance Tools Platform to show how the checklist supports and guides the testing process of these OSS products.

INTRODUCTION

Open Source Software (OSS) is currently enjoying increasing popularity and diffusion in industrial environments. However, verification and testing of OSS systems have not received the amount of attention they often have in Closed Source Software (CSS) (Zhao & Elbaum, 2003; Tosi & Tahir, 2010). Even very well-known OSS projects, such as Apache Httpd or the GCC compiler, do not seem to have mature testing processes and test suites defined inside their development process. This is probably due to a few mutually related reasons, which we outline.

DOI: 10.4018/978-1-4666-2937-0.ch004

1. Some testing techniques that are well established for CSS are not directly applicable to OSS systems, so a good deal of effort and cost is required for designing new testing solutions that are created *ad-hoc* for OSS systems.
2. OSS system development as a whole hardly ever follows the classic software engineering paradigms found in textbooks, so the execution of testing activities for OSS is less structured than for CSS.
3. The planning and the monitoring of the testing process of an OSS system hardly ever follow the guidelines used for CSS systems, so it is necessary to redefine some of the methods that are at the basis of the testing process.
4. OSS project testing often relies on the contributions from end-users, more than CSS does. This may occur via the distribution of testing or beta releases, or by using the feedback from users on stable releases. This allows OSS developers to shorten release cycles. In a way, this was stated by Linus Torvalds as "Given enough eyeballs, all bugs are shallow," which can be interpreted as referring to the importance of code review by peers, but also to the importance and power of testing by end-users contributing to OSS development with their feedback. The extent to which this occurs is larger than in CSS, and much larger than for products of other, more mature industrial sectors.
5. Often, OSS end-users independently test OSS products before using them. So, it is likely that the same kinds of tests are carried out over and over again by different users. This implies some wasted effort for end-users and lack of maturity of the testing process. In addition, this is against the very motivations of OSS, which implies sharing effort and reusing results, including those related to software verification and validation.
6. The lack of maturity of OSS testing processes is also indicated by the low coverage levels that are achieved by the test suites that are found along OSS products. Some results of the QualiPSo project show that the statement coverage levels of a number of well-known OSS projects do not exceed 25%. Clearly, this does not imply that at least 75% of source code statements have never been tested. Instead, this shows that a lot of the testing that is carried out goes undocumented, which is a symptom of low maturity.

Our experience in the context of OSS projects suggests that OSS communities do not usually view software testing as a primary software development activity. Also, most OSS projects do not fully integrate testing activities into their development process. In a survey, we asked 151 OSS stakeholders (developers, contributors, managers, end-users, etc.) to rate the importance of a number of factors that they take into account during the adoption of OSS components and products. The complete survey can be found in QualiPSo 1 (2010). It is worth noting that about 90% of the interviewees were actually involved in OSS projects as developers, contributors, integrators, managers, etc., and only 10% were end-users. Interviewees answered on average that the factor "existence of benchmarks / test suites that witness for the quality of OSS" has low importance. This may actually be a result of the fact that benchmarks and test suites are hardly ever available for OSS, more than the fact that benchmarks and test suites might not be important. So, OSS stakeholders do not use benchmarks and test suites simply because they often do not exist. We also analyzed the web portals of 33 well-known OSS products (Tosi & Tahir, 2010) and we discovered that, in the web portals

- Only 6% of the products provide the availability of a complete test suites
- 21% of the products provide performance benchmarks