Comparing Software Quality between Open Source and Closed Source Software Development

Jussi Heikkilä, Joona Hartikainen & Pramod Guruprasad
Abstract

1. Introduction

2. Software quality
   2.1 Definition
   2.2 Measurement
      2.2.1 Users’ perception
      2.2.2 ISO Standard

3. Current trends and discussions
   3.1 Users’ perspective
   3.2 Test suite analysis
   3.3 The Market Effect
   3.4 Open Source or Closed Source for My Project?

4. Software practices influencing software quality

5. Discussion

6. Conclusions

7. References
Abstract

Open source software development is a software development model wherein developers volunteer to take part in the development of a software product of their choice. The experience and credibility of the volunteering developers are often not accounted for in this model. Developers can simply join the discussion boards and mailing lists of the projects that they find they most interesting and start contributing towards its development. Most critics argue that this diversity in the development sphere of open source projects reduces the quality of the softwares produced under this model. This paper tries to compare the quality of the software product produced under open source and closed source software development methods. By analysing the quality of the end software product produced by the two methods of software development, we hope to provide a concise analysis on the possibility of success or failure in adopting open source software development.

Keywords: software quality, comparison, quality standards.

1. Introduction

Open Source Software (OSS) development is a school of software development that allows a group of enthusiast developers to come together to develop a software. The primary difference between open source and closed source school of software development is that the developers volunteer to be a part of the project. They do not necessarily belong to the company in charge of the open source project. As large number of developers who differ in working styles and methods of coding can come together to work on one project, maintaining superior levels of quality is a challenge.

By looking at the number of open source projects that are listed on websites such as SourceForge.net, GitHub.com and bug reporting websites such as Bugzilla.Mozilla.org, it is evident that open source software development is of interest to a very large number of developers (Groot, Kügler, Adams & Gousios, 2006). The interest shown by developers and non-developers in open source software development makes one believe that the quality of software produced by open source must be comparable to the quality of software developed under closed source development. It is believed that the software produced under closed source development methods are superior in quality based on the fact that the software is created by a dedicated team of developers. It is also believed that the product management team and the stakeholders execute the development in a structured and planned manner. This paper aims to understand the quality assurance and measurement metrics that are used in open source development by comparing it with the quality of software produced under closed source development. The research question for this article is “Does open source software development affect the overall quality of the product as compared to closed source software development?” Additionally, we will try to find out the reasons or differing processes affecting the quality
differences between closed and open source software.

For any software development community to work efficiently, it is important that there exist a sustainable community of developers (Aberdour, 2007). The biggest advantage in open source software development is that it facilitates for a large number of interested developers to partake in development. The developers can act as either a contributing developer, or a bug reporter, or even as an end user. This makes it possible for the core developers to focus on the primary aspect of the project. The contributing developers are expected to focus on only one functionality or feature such that it does not affect the work of the others. This is explained as the Onion Model of sustainable software development community by Mark Aberdour (2007). In the Onion Model (Figure 1), contributors to an open source project generally start from the outermost “User” layer. As time and requirements progress, Users can move to the next inner circle of “Bug Reporters” and then further onto “Contributing Developers”.

![Figure 1: The Onion Model](image)

© Mark Aberdour, 2007

This article is based on the analysis of the information available in articles, journals and publications on open source software development. The publications used as a reference for our research are based on empirical research or qualitative analysis that were conducted by the respective authors. Further, our own personal views and experiences are voiced to quantitatively analyse the information from these publications.

This report is presented with a total of six sections. Section 2 of this article explains the general meaning of software quality; Quality assessment metrics are also elaborated in the same section. A qualitative comparison of open source and closed source software development is presented in section 3. Section 4 lists a few reasons for the differences in the quality of the code
produced under open source and closed source. Our personal views on the quality of software produced under open source development is present in section 5 followed by the conclusion of the article in section 6.

2. Software quality

2.1 Definition

For a successful software project, the quality of the produced software is of utmost importance (Farooq & Quadri, 2011). Software quality can be expressed with definitions. Each definition emphasises on a certain metric that can be used to assess the quality of the software.

During the early stages of software industry, Barry Boehm defined software quality as “Achieving high levels of user satisfaction, portability, maintainability, robustness, and fitness for use” (as cited in Farooq & Quadri, 2011, p. 2). In the following year, as the importance of software industry was understood by a large group of industries, Carole Jones defined quality as “The absence of defects that would make software either stop completely or produces unacceptable results” (as cited in Farooq & Quadri, 2011, p. 2).

Over the period of years, the definition of software quality has evolved (along with the software development industry) to better incorporate users and developers alike. From the users' perspective, a software product that fulfills all their requirements and delivers results in a format that is apt to them is a software of good quality. (Petrasch, 1999)

As the usage of software products increased in new, previously unused fields, parameters such as flexibility, speed, ease-of-use, robustness, and portability are given more importance. (Kuan, 2002).

From these definitions it can be inferred that a successful software project has as few errors as possible while it is available for commercial use by users.

2.2 Measurement

2.2.1 Users’ perception

As indicated in the previous section, there are multiple ways to define the quality of a software project. These differences make it difficult to measure the quality of a software over just a few metrics.
Certain aspects of quality might be perceived to be more important by one group, whereas another group may define their own criterias to measure the quality of software. These differences can be attributed to their personal experience with software projects. Therefore, users’ perception is one parameter on which the quality of a software can be measured. (Bianco, Lavazza, Morasca, Taibi & Tosi, 2014)

Personal perceptions are of great importance in open source software development. Stakeholders often adopt open source solutions only if they perceive it as trustworthy. Trustworthiness of a software solution is based on the overall quality of the system (Bianco, et al., 2014). One way in which users and stakeholders assess the quality of the software is by understanding the structure of the organization building the software product. It is believed that the external quality characteristics of the software is directly related to the internal quality of the product. This relation enables us to assess the quality of the software by analysing the quality of the source code (Groot, et al., 2006).

As parameters such as software flexibility, speed, ease-of-use, robustness, and portability also play a major role in the users’ perception of the software product, it has to be noted that

Parameters such as software flexibility, speed, ease-of-use, robustness, and portability also play a key role in the users’ perception of the software product. These parameters are dependent on other factors such as hardware availability, execution speed of the hardware processor, and the likes. These external factors can have a strong impact on the user on their perception of the software product and lead them to believe that one product is far more superior than the other. (Kuan, 2002)

Another aspect that determines the quality of the product is the quality of the product team itself. The product team consists of all categories of the contributors, from users to core developers. Open source development has an edge over closed source development in terms of quality assessment. The availability of the source code and the transparency in the development process facilitates for third party assessment for quality. Further, all discussions related to the development process is publicly available on public forums such as discussion boards and mailing lists. Stakeholders can quickly gauge the quality standards of the software product by analysing the discussions and bug reports. The publicly available product development information can have a huge impact on the perception of the stakeholders on the software in review.

2.2.2 ISO Standard

In order to eliminate the differences in opinions on software quality the International Organization for Standardization formally introduced the ISO/IEC 9126. This standard has since been used to evaluate and measure software quality (Groot, et al., 2006). Figure 2 is a visual
representation of the ISO/IEC 9126 Software Quality Model along with associated parameters. In 2011, the ISO 9126 was replaced with a newer standard ISO/IEC 25010. ISO/IEC 25010 is called as the “Systems and Software Quality Requirements and Evaluation (SQuaRE)”. The new version is an extension of the previous ISO 9126 standard with more points added in order to keep up with the changing trends of software engineering and development.

A software can be pictured to be made up of several attributes such as functionality, usability, reliability, efficiency, portability, and maintainability. By closely examining the attributes that are listed by the ISO 9126, it is possible to enlist the defects and drawbacks of the software, if any exists at all. These attributes are the same for any software, irrespective of whether it was produced under the principles of open source or closed source software development, but the methods of evaluating the attributes are different. (Farooq & Quadri, 2011)

Software projects developed under open source model can be classified into two broad categories as:

1. “Projects that are developed to replicate and replace existing Contributed Source Software”.
2. “Projects initiated to create new software that has no existing equivalent Contributed Source Software”.

The emphasis on the software quality attributes that are listed in the ISO 9126 standard vary according to the type under which a software is classified. (Farooq & Quadri, 2011)

![Image: ISO 9126 Software Quality Model](https://example.com/iso9126-model.png)

Figure 2: ISO 9126 Software Quality Model

© ESSI-SCOPE, 1991
3. Current trends and discussions

In order to get a better picture of the current trends and discussions on maintaining quality in software projects, a few research publications and reports were studied. The studies and publications are handling software quality from different aspects and different methods are being used. Our idea is to show how the software quality is being affected in different situations. The analysis and findings of these research articles are presented below.

3.1 Users’ perspective

In order to better understand the users’ perception of the quality of products produced under open source, Bianco et al. (2014) conducted an empirical study in 2009. In their study, the participants rated 22 Java and 22 C++ projects which were developed under the principles of open source software development. The projects were rated for their perceived usability, reliability, security, functional requirements satisfaction, and portability.

The results of the survey found that the users perceived the quality of the projects to be equal, and in some cases better, than the projects developed under closed source development methodology. Only in 2 out of 32 cases were the closed source alternatives considered better. The choice of implementation language or the popularity of the project was not deemed significant in terms of trustworthiness of the product. (Bianco, et al., 2014)

3.2 Test suite analysis

Coverity Inc. is a free cloud based testing software which currently analysing about 6000 open source projects. The scanning tool can analyse and find defects for codes written in a variety of languages such as C, C++, C#, Java, and JavaScript.

In 2013, the Coverity Scan was used to analyse 250 million lines of C and C++ code from more than 740 open source projects. The scan report was published as the Coverity Scan Open Source Report later on. According to the report, the average defect density in the analysed C/C++ open source projects was 0.59 defects per every 1000 lines of code. This was an improvement from their 2012 report which indicated 0.69 defects per 1000 lines of code. According to Coverity Inc. (2014), a codebase with defect density of 1.0 or less can be seen as a sign of good quality.

To compare the results of the open source analysis with that from the closed source, Coverity Inc. analysed 500 million lines of code from 500 closed source C/C++ projects. The defect density for the closed source projects was a higher 0.72 defects per 1000 lines of code. The report indicates that open source software quality on average is better than the industry average
and improving. It must be noted that this conclusion was based on only those open source projects that used Coverity Scan for development testing.

3.3 The Market Effect

It is believed that the quality of a software system depends on the market it is competing in. (Raghunathan, Prasad, Mishra & Chang, 2005). A competitive market can be described as a market where in users or consumer select a certain product as superior by considering product values such as quality, value for money and ease of use. A common assumption is that, in a competitive market, the quality of the product goes up, while the associated costs are reduced. In a monopolised market, there is only one product for the users to use.

In their study Raghunathan et al. (2005) found the results to be surprisingly otherwise. For the study, the authors used game theoretic analysis to understand the dynamics of market driven quality of software products. Their report concludes by stating that software quality is not merely finding and correcting bugs but improving the functionality, maintainability, scalability, portability, and the usability of the software product as a whole.

In their analysis Raghunathan et al. (2005) found the following results:
- Quality of software produced under open source development is higher in monopolised market.
- In a competitive market, the quality of open source project is independent of closed source development.
- Quality of software produced under closed source development principles is affected negatively when competition from open source development increases.

These results can be attributed to various factors such as: in a monopolised market, the general quality of the open source software projects is better as the contributing developers are highly motivated. The motivation to produce good quality code that recognises their work and benefits them in their future work. On the other hand, in a competitive market, the quality is independent of each other as each school of development has different methods to incentivise the contributing developers.

The research also states that the software quality depends on other elements such as the number and quality of programmers, coordination between programmers, programmers’ efforts, number of users, incentive structure (tournament versus profit sharing).

Another study conducted by Choudhary and Zhou (2007) also found similar results as the study by Raghunathan et al. According to the study, competing against open source software in the same market can be very expensive and difficult for closed source products.
Choudhary and Zhou (2007) show in their research that competition from open source software can lower social wellbeing. The research states that when open source software contains high quality, it takes remarkable share of the market and reduces competing closed source software’s income. When closed source software’s income is reduced, it leads to a decrease in quality and lowers social wellbeing as well unlike in monopoly situation. This depends on how much it costs to the company to increase the closed source software’s quality.

The research shows that if the closed source software company lowers the price of the software to be able to compete with medium or high quality open source software, it affects the possibility to increase closed source software quality, because there is less money to be used. Closed source software’s quality will eventually drop when the cost of increasing quality is high and the closed source software is competing with open source software containing high quality. (Choudhary & Zhou, 2007)

3.4 Open Source or Closed Source for My Project?

As all the codes in the open source software are available openly for everyone, any person can contribute to the development of the software. Whereas, in a closed source software, the codes are accessible only by authorised personnels and any code improvements is possible only this dedicated team.

Khanjani and Sulaiman (2011) carried out a research in order to understand whether choosing open source software development methods would affect the quality of the software product, either positively or negatively. According to the research, the success of the software quality in closed source is managed by the members of the project, while in open source all the developers around the world are able to examine the code to identify and fix the issues and bugs in the code. This facilitates the open source code to be evaluated continuously by a large community. However, if a proper bugs and issues management system is not in place, multiple developers might work on the same bug fix.

The authors points out that the quality in open source software development model is being enhanced more than the quality in closed source software development because other tasks such as reviewing, testing, and maintenance in the development process of the software in open source are being performed simultaneously. They also highlight the fact that the open source method of development is more flexible, as in, it contains the ideas, shared knowledge and contribution of many developers.

Another advantage in open source development is in the process of identifying and listing the user requirements. This process appears to be simple and straightforward as the contributing developers themselves could also be the users of the product. (Khanjani & Sulaiman, 2011)
As previously mentioned, documentation and maintenance of a central code repository plays a major role in open source software development. Contributing developers have to be responsible for documenting their code such that it facilitates for easy maintenance and possible future improvements. Failing to document the code can single handedly contribute towards a substandard product. (Khanjani & Sulaiman, 2011)

In empirical study of open source and closed source software products the authors Paulson, Succi, and Eberlein (2004) found that open source software projects advances faster than closed source software projects. The authors analysis also pointed out that open source project is not necessarily more modular than closed source project, because open source software projects seemed to have problems with coupling.

The authors mentioned as well that defects in open source projects are being discovered and identified faster than in closed source projects. The research stated that open source software projects are not less complex than closed source software projects. (Paulson, et al., 2004)

4. Software practices influencing software quality

The development practices used in open source software development differ in many ways from the traditional commercial development. Although it appears to avoid some of the called best practices used in closed source development, it still manages to achieve software quality that is comparable. (Aberdour, 2007)

Common to large scale closed source development projects, quality is maintained through the practices of quality assurance and quality control. Quality assurance focuses on learning from mistakes and ensuring good management practices and it occurs throughout the organization whereas quality control is the process of verification and validation through high level plans and detailed structured testing process. (Aberdour, 2007)

Open source projects often lack this kind of quality model and maintaining quality comes from the modularity in the code base and the large community surrounding the project (Aberdour, 2007).

Open source software development relies heavily on the community built around it. Large open source projects have a large number of human resources at its disposal. Human resources of these levels cannot be achieved by a majority of the projects in closed source development. Most of these are users, who contribute by testing and reporting bugs. They play a key role in reducing the number of defects in open source projects which in-turn improves the overall quality of the code. Contributing developers also help fix bugs and review code, which allows the core developers to concentrate on more complex tasks or features without getting overwhelmed by bug fixes. The core team should be appropriately sized so that it can develop
the core functionality of the product as well as review and integrate high quality contributions into the central repository, in order to sustain the project. (Aberdour, 2007)

Stamelos, Angelis, Oikonomou & Bleris (2002) studied and measured quality of 100 open source applications written for Linux and found a direct relationship between high amounts of code modularity and software quality. They conclude that small component size and modularity is paramount in achieving low defect density and good quality specifically in open source software development. With more modular code, it is easier for many developers to work on independent parts, without affecting others. It also decreases the need to have intrinsic knowledge of the whole project which makes contributing easier for new developers.

Peer review, people management and testing are all common aspects of both closed and open source projects, but they differ in some important areas, which have a big impact on the code quality.

Peer review is, and has been, in use in closed software development for quite some time. But the availability of large numbers of contributing developers in open source projects allows for the code to be easily peer reviewed (Aberdour, 2007). The rapid release cycle that is frequently adopted by open source projects allow for a small, yet effective, peer reviews. This also helps keep the contributing developers interested in the project for a long term. An added advantage of peer reviews in open source development is that the contributing developers are constantly trying to outpace each other in finding or fixing bugs. This is in the interest of getting a recognition from the core developer team. A developer who has reviewed and fixed more code has a higher chance of being involved with the core development team in the future (Aberdour, 2007).

Between open source and closed source, open source development has access to larger human resource as compared to closed source. Although the quality of the available human resource cannot be gauged easily.

Because the community plays such a key role in open source projects, it is important for the developers to understand how to attract and sustain a large community around the project. As the number of people involved increases, so does the need for managing them properly. Just like closed source projects, large open source projects need to establish good coordination and workflow in order to keep the project from spiralling out of control. In open source development, a larger responsibility and the power to make final decisions is normally left to a few core developers. In order to be fair to the contributing developers, the decision process and discussion happens more openly and involves other contributors as well (Aberdour, 2007).

The methods of managing people differ from closed and open source projects. Open source development process is more focused on people than closed source software development which is more about methods and tools. In open source development, the work often happens more spontaneously through experimentation, but coordination and code improvements come at
a later stage unlike in closed source development which follows traditional development methods where importance is given to planning and execution. (Aberdour, 2007)

Software testing and quality assurance is another aspect which differs considerably between open source and closed source software development. In closed source development, the testing process is often a very rigorous process and follows structured testing methods and plans. In open source development most of the testing is done by the community. If the open source project lacks a large, committed and sustainable community, the quality will most likely drop drastically which again highlights the importance of a strong development and testing community around the project.

The results of a Linux kernel study show that 75 percent of all the work done for software project is typically mundane labor-intensive tasks like testing, code reviews and small bug fixes (as cited in Aberdour, 2007). Open source software projects excel in these kind of tasks, because of their large community with lot of human resources, whereas closed source projects typically have very limited resources at their disposal which might lead to such mundane tasks either getting overlooked and put into low priority or taking time from other important tasks.

5. Discussion

The popularity of open source school of development is constantly increasing in enterprise environments. In a study by Black Duck Software and North Bridge Venture Partners, 80 percent of the people who answered their survey chose open source software over closed source software based on software quality (as cited in Coverity Inc., 2007).

A common conclusion in all of the articles reviewed point out to the fact the there is no clear winner in terms of quality of the software developed. Most authors agree that the attributes and variables of the software quality needs to be identified in order to measure the quality of the product, irrespective of it being developed under open source or closed source development paradigm.

The studies show that both open and closed source software development methods can deliver equally high quality software, but the method of executing the actual project determines the quality of the end product. Choosing one over the other does not automatically mean better quality if the software quality attributes and variables are not taken into account.

Open source software development has emerged to be a more accepted approach for software projects as open source projects such as Linux, Apache HTTP server and MySQL were commended for the quality and became successful products. Open source software development has its own set of benefits such as developers around the world can participate into the project by searching for bugs, testing the project, evaluating, and fix issues in the
project. On the other hand, issues such as inefficient documentation, lack of a dedicated human resource management team can certainly impact the overall quality of the project.

The number of developers working with open source software project seems to have a big impact on software quality. Software quality has been noticed to increase when there are a lot of developers working on the open source software project. Software development with a lot of developers is able to use the large number of resources on bug fixing differently than closed source software bug fixing. It mainly focuses on fast bug fixing versus efficient bug fixing. For example, 10 developers can fix a bug in one working day, but total cost of fixing that bug is total of 10 working days. This option is fast but not very efficient and only open source software projects can use this option because closed source software projects lack the necessary number of resources. One developer fixes the same bug by spending 2 hours per day for one week and total cost of fixing that bug in this case is 14 person-hours. This option is not fast, but can be more efficient.

There are some aspects in open source software development which could be taken into closed source software development to help increasing software quality, for example: the tournament in open source software development where programmers are having competition between each other who writes higher quality code faster than others and this way gets some possible benefits in the future if the code gets merged into the actual open source project.

6. Conclusions

By analysing publications on software quality, we can now answer our research question - “Does open source software development affect the overall quality of the product as compared to closed source software development?”. The literature review helps us conclude that there is no direct effect on the quality of software produced under open source or closed source software development.

During the analysis, it was understood that the definition of software quality varied according to the viewpoint of the stakeholder. As there are multiple metrics on which software quality can be defined and analysed, it is extremely difficult to provide one comprehensive definition that caters to all the parameters.

It was found that the factors which have a definite impact on the quality of the software product are both external and internal. External factors such as competing market and users’ perception of the product play a major role in maintaining a high quality assurance of the product. Internal factors such as the sustainable development community and organization structure, irrespective of open source or closed source, play an important role in quality assurance.

While each of the development methods have its own set of pros and cons, the organization that is interested in creating the software product must take an informed decision in selecting a
development method. One advantage of open source development over closed source development in present day situation is the availability of a sophisticated and dedicated community of developers and testers.

Thus, it can be concluded that the quality of the project is independent of the development method selected. Metrics such as reliability, maintainability, and accuracy determine the quality and consequently the success of the software product.

7. References


