EVALUATING THE QUALITY OF FREE/OPEN SOURCE PROJECTS

Lerina Aversano, Igino Pennino and Maria Tortorella
Department of Engineering, University of Sannio, via Traiano 82100, Benevento, Italy

Keywords: Measurement, Documentation, Economics, Reliability, Experimentation, Standardization, Legal Aspects, Free/Open Source Projects, Quality Model, ISO/IEC 9126, ERP.

Abstract: Characterization and evaluation of software quality is one of the main challenge of software engineering. One of currently used standards is *ISO/IEC 9126*, which defines a quality model for software products. However, in the context of Free/Open Source software, differences in production, distribution and support modality, have to be considered as additional quality characteristics apart from ISO standard ones. This paper defines a quality model for Free/Open Source Software projects, equipped with an evaluation framework, realized by applying the *Goal Question Metric* paradigm. The evaluation of an open source system has been carried out as case study.

1 INTRODUCTION

Since many years, software engineering is facing software quality related problems. Lots of energy were spent for defining methodological and technological tools for managing such issues. The main requirement is the characterization of software quality and, consequentially, evaluation of the quality of a software system.

The International Organization for Standardization (ISO), faced the question by defining the ISO/IEC 9126 standard (ISO, 2004), published for the first time in 1991. It is a quality model for software products, to be considered as reference for evaluating them. Unfortunately, the ISO/IEC 9126 standard is not enough for characterizing the quality of an Free/Open Source software (F/OSS)[1] project. Actually, additional characteristics are required with reference to the global quality of the project, as a F/OSS project is different from a closed source one in terms of production, distribution and support modalities, more than product related characteristics.

Many organizations and researchers consider the evaluation of these aspects as necessary to assess the quality of an open source project.

In particular, Kamseu and Habra analyzed the different factors that potentially influence the adoption of an open source software (Kamseu, 2009). They identified a three dimensional model and stated that for having a good global project quality, it has to be considered the quality of: the development process; the community which made and maintain the product; and the product. Sung, Kim and Rhew focused on the quality of the product and identified some problems in evaluating an OSS product, such as the difficulty of using description and/or specification and collecting information if the developers do not make it public (Sung, 2007). IRCA (Wheeler, 2009) is an OSS selection process, based on side-by-side comparison of different software, defined by David Wheeler. The process consists of four steps: Identify candidates, Read existing reviews, Compare the leading programs' basic attributes to your needs, and Analyze the top candidates in more depth. The QSOS – Qualification and Selection of Open Source software – methodology consists of a set of steps regarding the start, evaluation, adjustment and selection OSS projects whose products seems to fit with the overall requirements (QSOS, 2006). The OpenBRR project – Business Readiness Rating for Open Source – born with the same purpose of QSOS's one (OpenBRR, 2005). QualiPSo – Quality Platform for Open Source Software – is one of the biggest initiatives related to open source software realized by the

[1] F/OSS stands for Free *libre* Open Source Software.

Aversano L., Pennino I. and Tortorella M. (2010).
EVALUATING THE QUALITY OF FREE/OPEN SOURCE PROJECTS.
In *Proceedings of the Fifth International Conference on Evaluation of Novel Approaches to Software Engineering*, pages 186-191
DOI: 10.5220/0003000701860191

European Union, and its products include an evaluation framework for the trustworthiness of Open Source projects (Del Bianco, 2008).

This research presented in this paper starts form the evaluation of the listed approach and proposes the EFFORT evaluation framework, aiming at overcoming their limitations. Then, the aim of the paper is:

- Definition of a quality model for F*l*OSS projects, extending the ISO/IEC 9126 standard and considering characteristics peculiar to that kind of projects.
- Definition of a framework for evaluating F*l*OSS projects, which gives guide lines, procedures and metrics to actually perform the measurement.

The paper is structured as follows: section 2 describes the proposed measurement framework; section 4 reports a case study, consisting of the evaluation of a F*l*OSS project; conclusions and future works are discussed section 5.

2 THE PROPOSED FRAMEWORK

This section presents the proposed evaluation framework, called EFFORT – **E**valuation **F**ramework for **F**ree/**O**pen sou**R**ce projects. Its main purpose is defining a quality model and measurement tool for supporting the evaluation of F*l*OSS projects, avoiding the limitation of the approaches analyzed in the previous section.

The quality model is synthesized in Figure 1. It defines the quality of a F*l*OSS project as the synergy of three major components: *quality of the product* developed within the project, *trustworthiness of the community* of developers and contributors, *product attractiveness* to its specified catchment area. Figure 1 shows the hierarchy of considered attributes.

The measurement framework was defined on the basis of the Goal Question Metrics paradigm. In correspondence of each first-level characteristics of Figure 1, one Goal is defined. Then, the EFFORT measurement framework includes three goals. Questions, consequentially, map the second-level characteristics, even if, Goal 1 has been broken up into sub-goals, because of its high complexity. For question of space, the metric level is not presented. The following subsections summarily describe each goal, providing a formalization of the goal itself, incidental definitions of specific terms and list of questions. A complete portion of the framework, with the questions, will be just shown for Goal 2.

2.1 Product Quality

One of the main aspects that denotes the quality of a project is product quality. So, it was necessary to consider all the aspects of software product quality, as defined by ISO/IEC 9126 standard (ISO, 2004).

Goal 1 is defined as follows:

*Analyze the **software product** with the aim of **evaluating** its **quality**, from a **software engineering**'s point of view.*

Given the vastness of the aspects considered by the ISO standard, Goal 1 is decomposed in sub-goals, each of which is focused on a single issue corresponding to one of the six main characteristics of the reference model: *portability, maintainability, reliability, functionality, usability,* and *efficiency*. The *in-use quality* characteristic is not considered in this context.

Table 1 shows the sub-goals and questions related *to portability, maintainability*.

For a precise definition of each characteristic, the ISO/IEC 9126 standard can be referred (ISO, 2004).

Table 1: Some sub-goals of the Product Quality.

Sub-goal 1a: *Analyze the software product with the aim of evaluating it as regards the portability, from a software engineering's point of view*	
Q 1a.1	What degree of adaptability does the product offer?
Q 1a.2	What degree of installability does the product offer?
Q 1a.3	What degree of replaceability does the product offer?
Q 1a.4	What degree of coesistence does the product offer?
Sub-goal 1b: *Analyze the software product with the aim of evaluating it as regards the maintainability, from a software engineering's point of view*	
Q 1b.1	What degree of analyzability does the product offer?
Q 1b.2	What degree of changeability does the product offer?
Q 1b.3	What degree of testability does the product offer?
Q 1b.4	What degree of technology concentration does the product offer?
Q 1b.5	What degree of stability does the product offer?

2.2 Community Trustworthiness

When adopting a F*l*OSS product, users are generally worried about offered support in case of troubles. The community, in fact, is not in duty-bound of supporting a user that adopts its software product. Anyway, a certain degree of support is generally given in quantity and modality that differ from a community to another one. We have considered

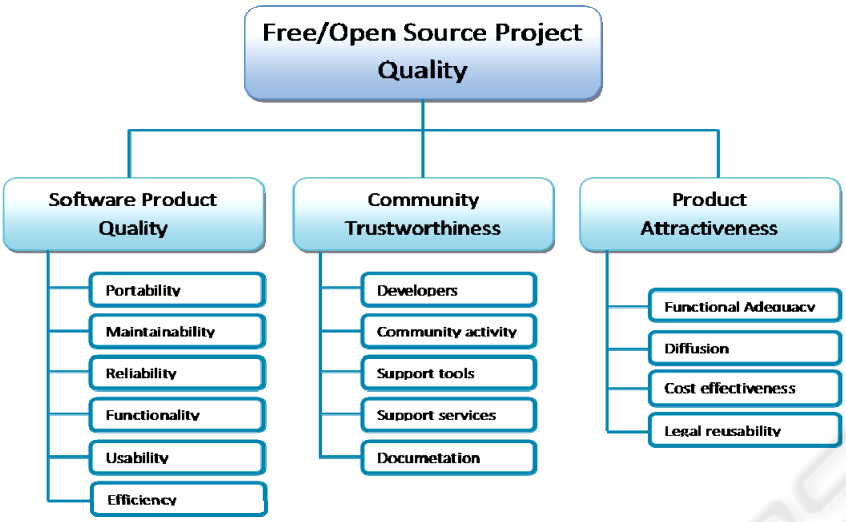

Figure 1: Quality model for F/OSS Projects.

valuable to include *community trustworthiness* in the definition of the global quality of a F/OSS project. With community trustworthiness, the degree of trust that a user can give to a community regarding the support. Goal 2 is defined as follows:

> *Analyze the offered **support** with the aim of **evaluating** the **community** with reference to the **trustworthiness,** from a **(user/organization) adopter**'s point of view.*

A community Generally provides a set of tools that support users in using its products such as forums, mailing lists, bug trackers, documentation, wiki and frequently *asking questions.* It is also possible to acquire a commercial edition of the software product, that usually differs from free edition in terms of support and warranties provided. Another important factor that influences trust in a project is the availability of documentation for installing, using and modifying the software product. All these aspects together with the activeness of the community are considered in the community trustworthiness concept. Table 2 shows the set of questions related to Goal 2. While Table 4 lists the metrics related to question 2.3.

Table 2: Questions about Community Trustworthiness.

Q 2.1	How many developers does the community involve?
Q 2.2	What degree of activity has the community?
Q 2.3	Support tools are available and effective?
Q 2.4	Are support services provided?
Q 2.5	Is the documentation exhaustive and easily consultable?

Table 3: Metrics related to question Q 2.3.

M 2.3.1	Number of thread per year
M 2.3.2	Index of unreplied threads
M 2.3.3	Number of forums
M 2.3.4	Average of threads per forum
M 2.3.5	Average of posts per year
M 2.3.6	Degree of internationalization of the forum
M 2.3.7	Number of trackers
M 2.3.8	Wiki volume
M 2.3.9	Number of frequently asked questions

2.3 Product Attractiveness

This goal has the purpose of evaluating the *attractiveness* of the product toward its catchment area. The term attractiveness indicates all the factors that influence the adoption of a product by a potential user, who perceive convenience and usefulness for achieving his scopes.

Goal 3, related to product attractiveness, is formalized as follows:

> *Analyze **software product** with the aim of **evaluating** it as regards **attractiveness** from a **(user/organization) adopter**'s point of view.*

This goal is more dependent from the application context than the other ones. The application context helps to explain why different kind of software products are developed. Two elements that have to be considered, during the selection of a F/OSS product, are functional adequacy and diffusion. The latter, in fact, could be considered as a marker of how the product is appreciated and recognized as useful and effective. This aspects are considered for formulating the questions of Goal 3 listed in Table 4.

Table 4: Questions about Product Attractiveness.

Q 3.1	What degree of functional adequacy does the product offer?
Q 3.2	What degree of diffusion does the product achieved?
Q 3.3	What level of cost effectiveness is estimated?
Q 3.4	What degree of reusability and redistribution is left by the license?

Concerning cost effectiveness, considered in Question 3.3, it is opportune to collect all the information regarding cost of services. The amount of available information can vary a lot among projects. For making the evaluation framework more complete with reference to a specific project, it is possible to add metrics whenever required. This can be also done also with reference to the license, referred in Questions 3.4. It can have a various degree of relevance, according to the purpose and needs of the users. In particular, the kind of license influences reuse and imposes some restrictions more or less severe regarding the possibility of including the code in own projects.

2.4 Data Analysis

Once data have been collected by means of metrics, it is necessary to aggregate them, according to the interpretation of the metrics, so one can obtain useful information for answering the questions. Aggregation of answers gives an indication regarding the achievement of the goals.

In doing aggregation, the following issues needs to be considered:

- Metrics have different type of scale, depending on their nature. Then, it is not possible to directly aggregate measures. To overcome that, after the measurement is done, each metric is mapped to a discrete score in the [1-5] interval.
- An high value for a metric can be interpreted in a positive or a negative way, according to the context of the related question. So, the appropriate interpretation is provided for each metric.
- Questions do not have the same relevance in the evaluation of a goal. A relevance marker is associated to each metric in the form of a numeric value in [1,5] interval. Value 1 is associated to questions with minimum relevance, while value 5 means maximum relevance.

A specific function that takes into account the observations above is so defined for the aggregation. Let us call with:

r_{id}, relevance associated to question *id* (sub-goal for goal 1);

Q_g, the set of questions (sub-goals for goal 1) related to goal g.

The aggregation function for Goal g is defined as follows:

$$q(g) = \left[\sum_{id \in Q_g} r_{id} * m(id)\right] / \sum_{id \in Q_g} r_{id}$$

where *m(q)* is the aggregation function of the metrics of question q:

$$m(q) = \left\{\sum_{id \in M_q} i(id) * v(id) + [1 - i(id)] * [v(id) \bmod 6]\right\} / |M_q|$$

where: M_q is the set of metrics related to question q; *v(id)* is the score obtained for metric *id* and *i(id)* is its interpretation. In particular:

$$i(id) = \begin{cases} 0 \text{ if the metric has negative interpretation} \\ 1 \text{ if the metric has positive interpretation} \end{cases}$$

3 CASE STUDY

For assessing the usefulness of the proposed method and identifying future works, a significant F/OSS project was evaluated by using EFFORT. The chosen project is **Compiere** (www.compiere.com), one of the most diffused ERP systems. Data were collected by analysing the documentation, trackers, repositories and official web sites of the project. In addition, the source code was analyses and the product itself was used. Further data source considered were *sourceforge.net, freshmeat.net* and *ohloh.net*.

In the following, data are reported in table and graphical format. The "in vitro" nature of the experiment did not allow a realistic evaluation of the efficiency, so it has been leaved out from the discussions of the results. In Table 6, one can observe that the Compiere product is characterized by more than sufficient quality. By analysing the sub-characteristics, one can notice that the product offers a good degree of portability and functionality, an excellent reliability and a sufficient usability. Concerning product quality results, the main limit of Compiere regards its maintainability.

Looking at reliability, the following consideration are recorded: a very good robustness, in terms of age, small amount of discovered post release bugs, low defect density, defect per module and index of unsolved bugs, and even higher recoverability, measured in terms of availability of backup and restore functions and services.

Table 5: Results regarding Product Quality.

Quality Characteristic	Relevance	Score
Portability	**3**	**4,1**
Adaptability		5
Install ability		2,64
Changeability		4,67
Maintainability	**3**	**2,83**
Analyzability		3
Modifiability		2,8
Testability		2,5
Technological Dispersion		3
Reliability	**3**	**4,42**
Robustness		4,16
Recoverability		4,67
Functionality	**5**	**4,13**
Functional Adequacy		3,25
Interoperability		5
Usability	**4**	**3,28**
Attractive		2
Operability		4
Comprehension		3,89
Learning ability		3,25
	Product Quality	***3,77***

Table 6: Results forCommunity Trustworthiness.

Quality Characteristic	Relevance	Score
N. developers	2	2
Community Activity	4	2,60
Support tools	5	2,44
Support services	2	3,44
Documentation	4	1,67
	Community Trustworthiness	***2,36***

Table 7: Results regarding Product Attractiveness.

Quality Characteristic	Relevance	Score
Functional Adequacy	5	3,25
Diffusion	4	4
Cost Effectiveness	3	2,40
Legal Reusability	1	5
	Product Attractiveness	***3,63***

As Compiere is an ERP software, the presence of a transaction management systems could also be considered. Concerning maintainability, the lower score has been evaluated by mainly using CK metrics (Chidamber,1991), associated to the related sub-characteristics. For instance, the medium-low value for testability of Compiere depends on the high average number of children (NOC) of classes, number of attributes (NOA) and overridden methods (NOM), as well as little availability of built in test functions. The values of cyclomatic complexity (VG) and dept of inheritance tree (DIT) are on the average.

Table 6 reports data regarding the community trustworthiness. In this and the next cases, the hierarchy of characteristics has one less level.

The score obtained by Compiere for community trustworthiness is definitely lower than the product quality. In particular, community behind Compiere is not particularly active; in fact, average number of major releases per year, average number of commits per year and closed bugs percentage are low values. Support tools are poorly used. In particular, a low activity in official forums was registered. Documentation available free of charge is small; while support by services results to be more than sufficient, even if it is available just for the commercial editions of the product. This aspect reflects the business model of Compiere Inc., that is slightly distant from traditional open source model: product for free, support with fee.

Figure 2 shows a graphical representation of the results. Looking at Table 7 and Figure 3, one can notice that Compiere offers a good global attractiveness. In particular, a sufficient functional adequacy and an excellent legal reusability is exhibited, because of the possibility left to the users of choosing the license, even a commercial one. Compiere does not seem to be very affordable, compared to other F/OSS solutions. Compiere's product results quite diffused. The last characteristic was evaluated by measuring: number of downloads, index of freshmeat popularity, rating number of sourceforge users, rating index of positive sourceforge, number of success stories, visibility on google, number of official partners, as well as number of published books, experts review and academic papers.

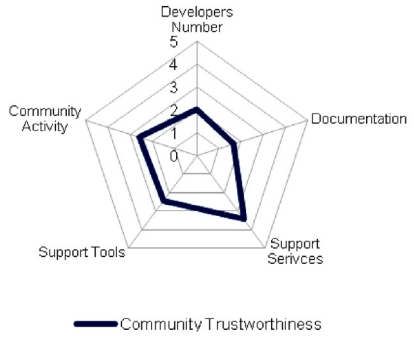

Figure 2: Compiere Community Trustworthiness.

4 CONCLUSIONS

The work presented in this paper was motivated by the necessity of having tools and models for characterizing and evaluating the quality of F*l*OSS projects, comprehensive of quality characteristics of the product and peculiar aspects of such a kind of projects.

The proposed study started by analysing many approaches already existing for evaluating F*l*OSS. All the considered approaches presented limitations for performing a complete evaluation. Among them, IRCA seems to be the most complete (Wheeler, 2009), but it did not include an evaluation framework to perform the measurements. EFFORT overcomes this limitation by proposing a measurement framework that is directly applicable.

EFFORT was designed for completely covering the intersections among the other analyzed approaches. It offers a good coverage of the ISO/IEC 9126 standard, with the exception of in-use quality. Other characteristics analyzed by the other approaches and considered significant were also considered, such as: QSOS' maturity, pretty much covered by EFFORT's diffusion; cost effectiveness and OpenBRR's Architecture, of which EFFORT considers just dependence of third parts components.

During the analysis of the case study regarding the Compiere project, it was noticed that some characteristics of the ERP systems were not considered by EFFORT. In particular, the configurability and customizability of such a kind of systems. In particular, they could be considered in the context of attractiveness. This aspect suggests an evolution of the EFFORT approach that considers a specialization of the measurement framework to the specific peculiarities of a F*l*OSS projects before its application. Therefore, future work will regard the definition of mechanisms for extending and customizing EFFORT, and offering the possibility of a better characterization of all aspects dependent on the application domain.

REFERENCES

ISO -International Organization for Standardization, 2001-2004. ISO standard 9126: Software Engineering – Product Quality, part 1-4. ISO/IEC.

Chidamber, S. R., Kemerer, C. F., 1991. Towards a metrics suite for object-oriented design". In *ACM SIGPLAN Notices*, 26(11), ACM press, pp. 197 -211.

Kamseu, F., Habra, N., 2009. Adoption of open source software: Is it the matter of quality? PReCISE

Sung, W. J., Kim, J. H., Rhew, S. Y., 2007. A quality model for open source selection. In *ALPIT 2007*, *Sixth International Conference on Advanced* Language Processing and Web Information *Technology* IEEE Comp. Soc. Press.

Wheeler, D. A., 2009. How to evaluate open source software/free software (OSS/FS) programs, DOI= http://www.dwheeler.com/oss_fs_why.html

QSOS, 2006. Method for Qualification and Selection of Open Source software. Atos Origin.

OpenBRR, 2005. Business Readiness for Open Source. Intel.

Del Bianco, V., Lavazza, L., Morasca, S., Taibi, D., 2008. The observed characteristics and relevant factors used for assessing the trustworthiness of OSS products and artefacts. QualiPSo.