

Quality of Open Source Software: the QualiPSo Trustworthiness Model

Vieri del Bianco¹, Luigi Lavazza¹, Sandro Morasca², and Davide Taibi²

1 Università degli Studi dell'Insubria, Dipartimento di Informatica e Comunicazione,

Via Mazzini, 5 – 21100 Varese, Italy

{delbianco,lavazza}@uninsubria.it,

WWW home page: <http://www.dicom.uninsubria.it>

2 Università degli Studi dell'Insubria, Dipartimento di Scienze della

Cultura, Politiche e dell'Informazione

Via Valleggio, 11 - 22100 Como, Italy

{sandro.morasca, davide.taibi}@uninsubria.it

WWW home page: <http://dscpi.uninsubria.it>

Abstract. Trustworthiness is one of the main issues upon which the decision whether to adopt an Open-Source Software (OSS) product is based. The work described here is part of an activity that has the goals of 1) defining an adequate notion of trustworthiness of software products and artifacts and 2) identifying a number of factors that influence it. Specifically, this paper reports about the identification of the “dimensions” of trustworthiness, i.e., of the high-level qualities that software products and artefacts have to possess in order to be considered trustworthy. These dimensions are described by means of a conceptual model of trustworthiness, which comprises the representation of the factors that affect the user’s perception of trustworthiness, as well as the objective characteristics of the products that contribute to “build” trustworthiness. The aforementioned model is equipped with a measurement plan that describes, at the operational level, how to perform the evaluation of the trustworthiness of OSS products. The proposed model provides the basis to build quantitative models of the trustworthiness of OSS products and artifacts that are able to explain the relationships between the (objectively observable) characteristics of OSS products and the level of trustworthiness perceived by the users of such products.

1 Introduction

The trustworthiness of a software product is one of the main aspects that contribute to its adoption/rejection. This is true for any software product, but it is especially true for OSS products, whose trustworthiness is sometimes still regarded by some as not as guaranteed as that of closed source products, or viewed as more difficult to assess. Only recently have many industrial organizations started investigating the potential of OSS products as users or even producers. As they are now getting more and more

involved in the OSS world, these software organizations are clearly interested in ways to assess the trustworthiness of OSS products, to choose OSS products that adequately fit their goals and needs. To help foster the adoption, use, and production of OSS products, it is therefore important that the real goals and needs of software organizations be given top priority when investigating assessment methods, techniques, and indicators for OSS products.

The definition of a method to assess the trustworthiness of OSS products is one of the main goals of the QualiPSo (Quality Platform for Open Source Software) project, which is an ongoing initiative funded by the EU with the aim of investigating the trustworthiness of OSS, to identify its weaknesses, strengths, and possible improvement areas and ultimately improve its quality so as to support and promote its acceptance. More on QualiPSo can be found at <http://www.qualipso.org>.

Within the QualiPSo project, the trustworthiness of OSS products is addressed through several activities. In this paper, we describe a new model of OSS product trustworthiness that allows us to provide definitions for the dimensions of trustworthiness that are unambiguously understood and on which a widespread consensus can be achieved. This is a necessary step, since all too often in Software Measurement there is a lack of agreement about the real meaning of a number of software qualities. Based on these dimensions, a set of metrics are defined to capture the various components of trustworthiness from different viewpoints.

The work reported here is organized in two phases.

In the first phase, the various dimensions of trustworthiness of software products and artifacts are identified and described, on the basis of the results of a survey of the perception of trustworthiness in the European industry [14]. These dimensions of trustworthiness represent the point of view of the user, i.e., *the user's perception of trustworthiness*. While previous work [14] provided various indications concerning the qualities that underlie the perception of OSS trustworthiness, we define a framework for classifying these qualities and establishing their contribution to the notion of trustworthiness as precisely as possible. The result is a (semi-formal) model of trustworthiness, i.e., a hierarchy of qualities and sub-qualities whose ensemble composes the notion of trustworthiness. The conceptual model was also refined into a measurement plan that describes how to quantitatively evaluate each element of the conceptual model.

The second phase of the work concerns the definition of:

- A conceptual model of the trustworthiness of OSS products: this model defines trustworthiness in terms of the product's qualities, approximately like the ISO 9126 standards define the qualities of software. In our model, the trustworthiness of OSS products depends on qualities like As-is utility (quality in use), Exploitability in development (that is, how well the OSS product can be used for the development of other products, e.g., through integration or customization), Functionality, Interoperability, Reliability, etc.
- A model of the trustworthiness of OSS products that captures the factors that influence trustworthiness. This model is more detailed than the conceptual model mentioned above: it relates the qualities that compose trustworthiness to the characteristics of the OSS products. For instance, the Exploitability in

development quality depends on the Maintainability of the product, which depends on sub-qualities like Analyzability, which, in turn, depends on characteristics like product modularity.

- A set of measures that quantify both the factors that compose trustworthiness and the product characteristics that influence the trustworthiness qualities.

These models provide definitions of unambiguous product characteristics, to provide an objective base for evaluating (or estimating) trustworthiness.

The measures defined here will be used in the construction of a mathematical model that relates the qualities of OSS product to the corresponding level of trustworthiness perceived by users.

The following example may help clarify the difference between the work performed in the two phases mentioned above. The user's perception of trustworthiness is determined –among other properties– by the efficiency of the product. Of course, the importance of efficiency depends on the product and the user; similarly, the perception of efficiency varies from user to user: for instance, one user may be perfectly satisfied with the efficiency of a product while another user is not very satisfied (e.g., because he/she uses the product in a different way or context). The first phase deals with the definition and measurement of the various aspects of trustworthiness as perceived by the users. On the contrary, efficiency (and the other properties that underlie the notion of trustworthiness) can be linked to objectively definable and measurable characteristics, e.g., how long it takes to perform a set of typical tasks using the product, how many resources are consumed for average usage, etc. The second phase deals with the definition and measurement of the various characteristics of products that determine the product's properties that contribute to its trustworthiness. We expect that the subjective perception of trustworthiness depends on the objective characteristics of the product: this relation will be explored in future work.

In this paper, we report the construction of the aforementioned trustworthiness model, and the first steps of the definition of the corresponding measurement plan. We plan to use the framework reported here to measure and evaluate several OSS products: the collected data will be analyzed to quantitatively validate the existence of a correlation between the user's perception of trustworthiness and the objectively measurable characteristics of the OSS. Such a correlation would provide extremely valuable indications concerning the properties that OSS products should have in order to be trusted by the users.

2 The proposed approach to modelling trustworthiness

In the construction of the models, the dependencies and relations described in Figure 1 were taken into account as follows:

- Trustworthiness is a property that relates to a product.
- A product has a set of intrinsic qualities (modularity, complexity, size, etc.).

- The trustworthiness of a product depends on the perception that users have of the qualities (Functionality, Reliability, Performance, etc.) of the product.
- The user’s perception of the qualities depends on the intrinsic qualities of the product.
- The user’s perception of the qualities also depends on the type of usage.

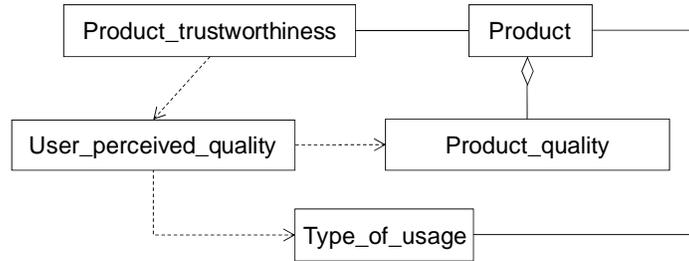


Fig. 1. Trustworthiness: dependencies on the users and the product’s characteristics

Therefore, a model of trustworthiness can be characterized according to two characteristics: the nature of the model (which can be conceptual or operational) and the point of view (i.e., how is the trustworthiness evaluated).

In our work, we consider two points of view:

- The user’s point of view: he/she evaluates the overall trustworthiness of the product, as well as the individual qualities that affect trustworthiness. For instance, a user will report the Functionality or the As-is utility of the product according to his/her needs and type of usage.
- The “objective” point of view, which addresses the evaluation of intrinsic qualities of the product and their role with respect to trustworthiness.

The user-dependent and the objective, software-dependent views are clearly related: for instance, the individual qualities that underlie trustworthiness (e.g., functionality, reliability, etc.) are the same in both models.

The proposed model includes a conceptual part, which describes the meaning of the qualities, and an operational part (the measurement-oriented one), which provides enough details to support the quantitative evaluation of the various aspects of trustworthiness. The latter part is defined by means of the GQM technique [1][2].

The result is a unique GQM plan, which includes both the user’s perspective and the objective evaluation.

The GQM goal in this GQM plan is actually a meta-goal. It must be instantiated into several goals, one for each OSS product analyzed. The detailed definitions of the characteristics that affect trustworthiness depend on the product, so some measures in the GQM plan are defined in a relatively abstract manner: once a decision is reached on the product to which the GQM plan is applied, those measures are precisely defined.

3 Software quality models: existing approaches and new ideas

The notion of trustworthiness is inherently subjective, because trustworthiness requirements depend on how the software is used, and because different users evaluate an OSS product according to different criteria and points of view.

Therefore, in order to assess the trustworthiness of software, it is not wise to look for a general and ubiquitous set of characteristics and parameters; instead, we should define and apply an evaluation process that is tailored to the requirements the software has to fulfil, and that takes into consideration the points of view of multiple users. This approach should not be surprising: in empirical software engineering, no single, one size-fits-all set of measures exists that can be used for all products, environments, and goals.

Although it is commonly agreed that trustworthiness encompasses the reliability, security, and safety of a software system [3][4], as well as fault-tolerance and stability, a trustworthy software product has to possess additional qualities.

Whether OSS is more or less trustworthy when compared to similar proprietary software is still a matter of hot debates and controversial opinions. Even though some believe that OSS is intrinsically at least as trustworthy as proprietary software [6], there are opinions pointing to the opposite ends of the spectrum: from OSS enthusiasts [7] [6] to much more cautious and sceptical viewpoints [5], there exists a complete range of perceptions of the trustworthiness of OSS.

The subjectivity of the evaluation, the different points of view, and the ethical, economical, and political issues concerning the adoption of OSS call for a rigorous and technically sound approach to the evaluation of OSS trustworthiness. This is the starting point of the work reported here: we need to understand the dimensions of trustworthiness, the roles of the individuals involved in trustworthiness evaluation, the problem domain addressed by the software product whose trustworthiness is evaluated, and the relationships between these aspects.

We do not need to start from scratch, since a huge amount of work on software quality was done in the past. The best known source of indications about software quality is the ISO 9126 software quality model standard [8]. The first of the ISO 9126 standards, namely ISO 9126-1, defines a Quality Model via a set of quality characteristics and sub-characteristics, as shown in Fig. 2.

The ISO 9126-1 standard uses qualities that were believed to be the most relevant ones when the standard was defined.

Recently, there is the tendency to add security and interoperability to the set of ISO 9126 qualities, as recognized in the new set of ISO 25000 standards. Security and interoperability are already present in the ISO 9126 standard, but only as “sub-factors” of functionality.

Our proposal draws upon the existing proposals for software quality evaluation models, but we need to focus on the concept of trustworthiness, which is a multi-faceted quality, and on OSS. However, the existing proposals for the evaluation of OSS tend to be rather narrowly focused, so adopting one or more of them may lead to an incomplete or unbalanced evaluation.

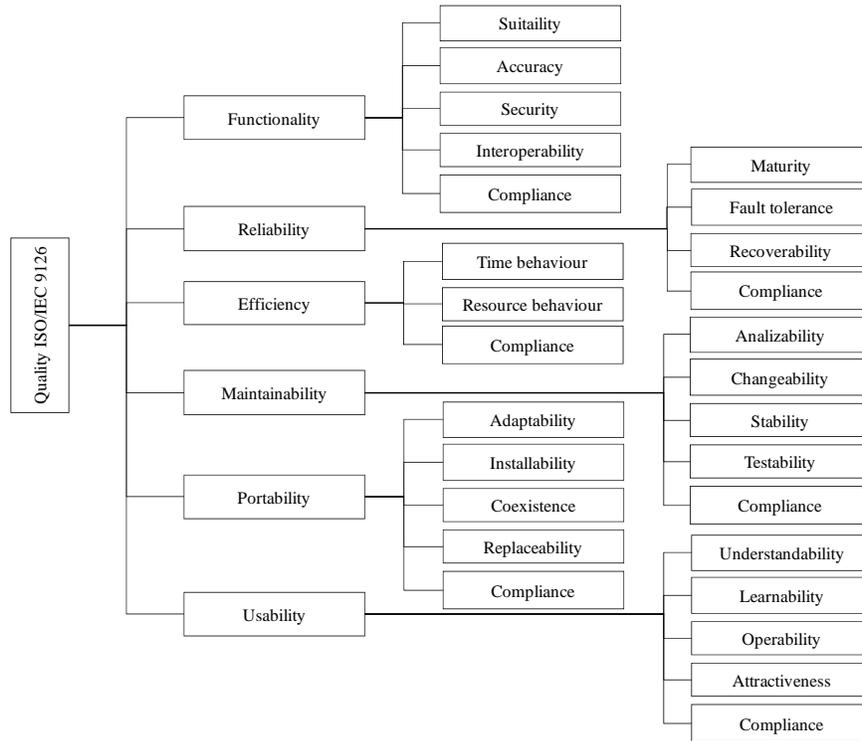


Fig. 2. The ISO 9126 quality model

The TCO (Total Cost of Ownership) [11] addresses the evaluation of the cost of adopting and using a software program, including all the expenses, and spanning the whole lifecycle of the system. Although TCO has the merit of providing a comprehensive basis for the evaluation of SW costs, it is limited on two important issues: it does not address the evolution of the OSS user’s process, which could require updating the software; it provides only a partial view of the cost effectiveness of OSS, since it ignores the evaluation of the full set of benefits of OSS.

Other methods focus on the quality of OSS and its contribution to the user’s business goals. For instance, the Open BQR [11] is based on the assessment of a number of relevant aspects of an OSS product, including: Functional adequacy to requirements, Quality (in terms of absence of defects or time-to-fix), Availability of maintenance support, etc. Open BQR is based and extends other methods for the evaluation of OSS, like OSMM (Open Source Maturity Model) [17], QSOS (method for Qualification and Selection of Open Source software) [18] and Open BRR (Business Readiness Rating for Open Source) [19].

4 Overcoming the limitations of existing approaches

The methods mentioned in the former Section concentrate on technical issues; none of these addresses issues like the cost of the adoption or the adaptation/extension process. Thus, the available techniques do not provide a complete and balanced view of the OSS contribution to the user's business. For this purpose, it is interesting to look at the Balance Scorecards method [13].

The Balanced Scorecards (BSCs) technique is a measurement-supported strategic management method for general purpose organizations (i.e., not specifically for ICT organizations). It was proposed in the early 90's to overcome the limits of traditional management-oriented metrics (e.g., the Return On Investment) that were too centred on a financial view of organizations and were limited in scope (in that they provided an all-internal view of a company situation) and time (they concerned only the past performance of the company).

To obtain a more complete and effective view of the state of an organization, it was proposed to measure, in addition to financial issues:

- The performance towards the outside world: *customer* satisfaction was considered the most representative indicator.
- How well the organization is equipped to be successful in the *future*. The ability to innovate, learn, and grow is considered a fundamental domain of the BSC method.
- The performance of the *internal process*, which is directly linked to customer satisfaction and financial results, and that is where the learning and growth take place.

A few years ago, the BSC approach was adapted to ICT, to provide the ICT departments of large companies with a tool to measure the contribution of ICT to the main business of the company in a complete and balanced way, thus overcoming the traditional view of ICT as a cost [15]. Here, we are concerned with applying BSC to OSS. A first proposal on this is reported in [16].

Considering again the TCO and Open BQR techniques in the framework of the BSC, it is quite clear that they do not provide a complete and balanced evaluation. The Balanced Scorecards technique suggests that in order to evaluate an OSS product's trustworthiness we consider also how well the OSS product contributes to the business process of the user, how well the OSS product supports the user organization in addressing changes and new challenges, how the usage of the OSS product contributes to the perception of the organization from outside (e.g., by customers).

To show that applying BSC to OSS evaluation may be useful, let us consider the following example. Suppose that an organization decides to adopt an OSS product instead of buying the licenses for using an equivalent commercial product.

The first, obvious effect of this decision is that the license costs disappear and the commercial software becomes unavailable to the organization. Both effects can be precisely classified in the BSC framework. The beneficial effect of not paying the

licence is accompanied by negative effects in all the other sectors: the process is no longer supported by the software applications, which need to be replaced; as a consequence, from the customer point of view quality and support issues arise; finally, from the growth perspective, maintenance and support issues arise.

Note that we here indicate only the qualitative effects of the decision, but according to the BSC we should define proper measures for a quantitative evaluation of different issues: from costs of licences, to the efficiency of the process, to the quality of the products, to the satisfaction of the customers.

The second part of the decision is that OSS is used. These effects too can be precisely classified in the BSC framework. The evaluation shows that:

- From the financial point of view, OSS is not for free: the organization will have to adapt it, configure it, and possibly perform maintenance activities.
- From the point of view of the process, the OSS is suitable, and with respect to some issues even better. This is quite common with OSS: having the possibility to instrument the code means better testing of functionality and security.
- From the learning and growth perspective, we have a negative effect (the cost of learning) and a positive effect (the knowledge of the software allows faster and better responses to new requirements).
- From the customer perspective, being recognized by the OSS community as a qualified user and/or developer of OSS increases the reputation of the organization.

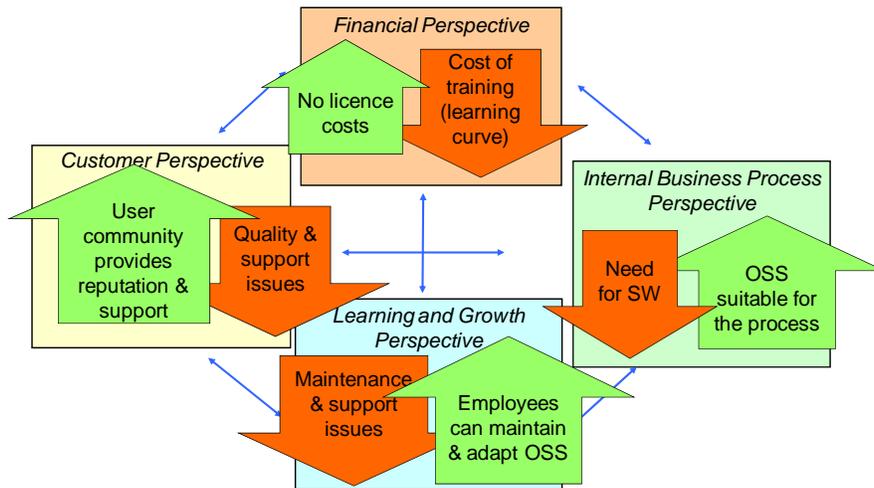


Fig. 3. An example of BSC: effects of adopting Open Source software

Finally, we have to combine the effects illustrated to get the complete picture, illustrated in Fig. 3. The measurement of the various aspects may prove that the effects of the decision are balanced, and the global consequences of the decision

match the organization's goal. In this case, we would find that the license savings are partially compensated by the need to adapt and configure the software, and that the lack of the (supposedly high-quality) commercial software is compensated by the ability to configure and adapt the OSS in a more timely and effective manner.

The content of the example described above cannot be generalized to whatever situation. For instance, it is not always the case that "Quality & support issues" arise, which have a negative influence from the customer perspective. The point is that the balanced scorecard method helps taking into consideration several (if not all) of the issues that are relevant for a correct evaluation of the prospected situation.

5 The definition of the measurable model

According to the observation reported in Sections 1 and 2, the GQM plan needs to include questions concerning the OSS product, the users and uses, the developer and the perceived qualities. To get the most complete and reliable model of trustworthiness, we address both subjective (i.e., user dependent) and objective (i.e., measurement-based) evaluations.

The GQM plan presented here consists of a single goal. This is a very general goal that does not strive to focus on specific aspects or situations, at the cost of including a large number of questions and metrics. The goal is defined as follows:

Analyse OSS for the purpose of evaluating/estimating the trustworthiness from the point of view of OSS users and developers in "business" organizations.

The goal mentions business organizations, as we are interested in the adoption of OSS in environments (like industry and the Public Administration) where the usage of OSS can have a financial/economic impact.

The proposed goal adopts a generic point of view, which includes both the developers and the different types of end-users.

According to the findings of [14], the indications of the literature and the standards, and the considerations reported above concerning the need for a balanced and complete evaluation, it seems reasonable to define trustworthiness according to the qualities schematically reported in Fig. 4 and described below.

As-is utility (quality in use) is the quality that the users seek when they want to use the OSS product "as-is", i.e., without changing the code.

Exploitability in development indicates how easy, efficient, effective, etc., it is to change, maintain, and develop the product, possibly to include it into another product.

Functionality indicates the degree to which the considered OSS product satisfies/covers functional requirements. This quality, as well as the following ones is desirable in general, i.e., both if the product is used as-is, or if it is changed.

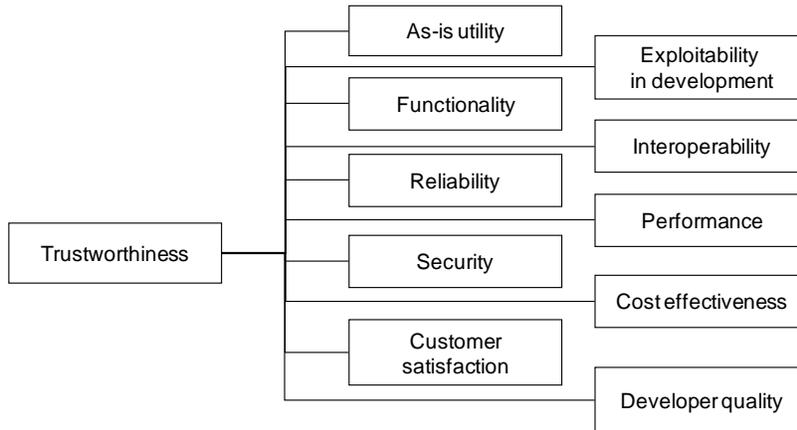


Fig. 4. The model of the perceived trustworthiness

Interoperability indicates how well the OSS product operates in conjunction with (i.e., exchanging data or control information with) other software products. Note that sometimes it is not easy to distinguish functionality and interoperability. We tend to consider the “interactions” that are explicitly required as functionalities, while interoperability deals with unanticipated situations. For instance, a compiler is explicitly required to produce an output that can be read by an interpreter (either hardware or software), but nobody would speak about the interoperability between the translator and the consumer of the translation. Another program may be required to produce a report or a log of activities: interoperability is concerned with reading that report through another product (a feature in which not all users are interested!).

Reliability indicates the ability of the software not to fail, i.e., to perform its function satisfactorily.

Performance indicates the ability of the software to perform its function within given constraints concerning the consumption of resources and time.

Security indicates the ability of the software to prevent unauthorized access to programs or data.

Cost effectiveness indicates the ability of the software to contribute positively to the financial balance.

Customer satisfaction indicates the ability of the software to contribute positively to satisfying the customer (i.e., the final beneficiary of the process in which the OSS product is involved).

Developer quality (developer reliability) indicates (indirectly) that we can expect a reasonably good quality of the current version of the product, and regular maintenance and evolution of the product.

Table 1 summarizes the differences among the top-level trustworthiness qualities defined in this WP and the corresponding qualities considered in [14] and in the ISO 9126 standard.

Table 1. Trustworthiness qualities here, in Qualipso survey [14] and ISO 9126

Quality name	Qualipso survey [14]	ISO 9126
As-is utility	Only some sub-qualities present	✓ (Quality-in-use)
Exploitability in development	Only some sub-qualities present	Only some sub-qualities present
Functionality	✓	✓
Interoperability	✓	✓ (sub-factor of functionality)
Reliability	✓	✓
Performance	(implicitly addressed as part of functionality)	✓ (efficiency)
Security	✓	✓ (sub-factor of functionality)
Cost effectiveness	✓	(addressed only partly by productivity)
Customer satisfaction	✓	(addressed only indirectly by user satisfaction)
Developer quality	✓	✗

It is possible to see that the trustworthiness qualities defined here match quite closely the factors described in [14], with the difference that here we have tried to structure the model of trustworthiness around the qualities that the users are presumably more interested into. Accordingly, we have highlighted the two typical types of usage of OSS products: as-is use and modification/development based on OSS products. These two types of use give rise to specific quality perspectives: As-is utility and Exploitability in development. Since these qualities are specific of OSS products, quite naturally they match only partially the ISO 9126 qualities.

As described in Section 4, it is desirable that the qualities address all the perspectives of the Balanced Scorecards: accordingly, we have made our definition of trustworthiness complete and balanced by introducing a few specific factors, such as Cost effectiveness and Customer satisfaction.

The structure above does not need to be reflected very faithfully in the GQM plan. It is more of a guideline for assuring the completeness of the plan and for guiding the data interpretation process. In fact, the GQM plan is organized as follows:

- A first part of the plan is dedicated to capture information about the identity of the product and the producer.
- A second part is dedicated to collect the user perception of the trustworthiness of the product, both at a global level and at the level of qualities (e.g., functionality, reliability, modifiability, etc.). The evaluations described here are intrinsically subjective, since they reflect the point of view of the user.
- The third part of the plan is dedicated to identify the characteristics and properties of the product that are believed to contribute to the user's perception of trustworthiness. In this section, the objective properties of the OS products are

identified. Actually, when objective measures are not feasible, they can be replaced by equivalent subjective evaluations.

Table 2. The abstraction sheet of the GQM plan

<u>Object</u> OSS	<u>Purpose</u> evaluate/estimate	<u>Quality Focus</u> trustworthiness	<u>Viewpoint</u> OSS users and developers	<u>Environment</u> “business” organizations
		<u>Quality Focus</u>	<u>Variation Factors</u>	
		ID_OSSproduct ID_User_Info ID_Developer User_Trustworthiness Q_User_As-is utility (quality in use) Q_User_Exploitability_in_development Q_User_Functionality Q_User_Interoperability Q_User_Reliability Q_User_Performance_Resources Q_User_Performance_Time Q_User_Security Q_User_Customer_Satisfaction Q_User_Cost_Effectiveness Q_User_Developer_Quality(reliability) Q_Actual_As-is utility (quality in use) Q_Actual_Exploitability_in_development Q_Actual_Functionality Q_Actual_Interoperability Q_Actual_Reliability Q_Actual_Performance Q_Actual_Security Q_Actual_Developer_Quality(reliability)	CodeCharacteristics	
		<u>Baseline Hypotheses</u>	<u>Impact on Baseline Hypotheses</u>	
		Baseline hypotheses are given by the results of [14].	The consequences of variations on the B.H. are documented in the literature.	

The abstraction sheet of the GQM goal is reported in **Table 2**. It is possible to see that the names of several quality foci in the abstraction sheet above start with “Q_user”. These quality foci represent the user’s perception of trustworthiness.

The quality foci whose name starts with “Q_actual” represent the qualities that are relevant to trustworthiness evaluated from an objective (i.e., user independent) point of view. These are the factors (mainly OSS product qualities) that are expected to affect trustworthiness: they are identified and characterized by measurable attributes.

Both the “Q_user” and “Q_actual” quality foci are being expanded into questions and metrics in the context of the work done in the QualiPSo project. For space reasons, we cannot give details here; however in Fig. 5 we illustrate the refinement of the “non traditional” qualities of our trustworthiness model.

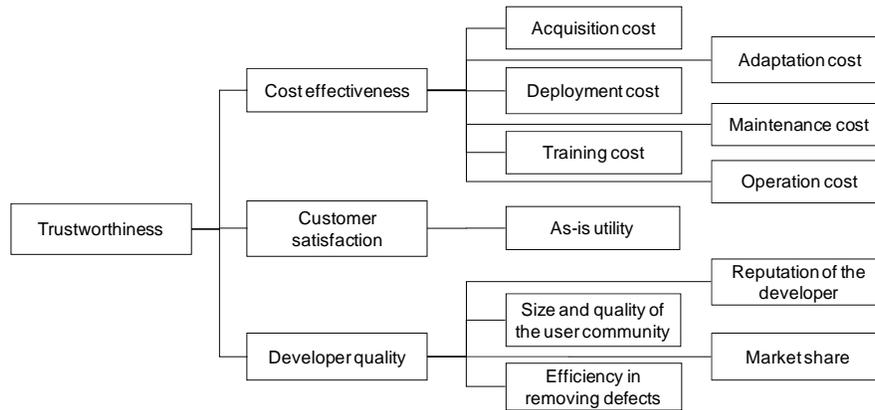


Fig. 5. The model of the perceived trustworthiness

6 Conclusions and future work

Being able to evaluate the trustworthiness of OSS is fundamental for two very important purposes. On the one hand, a reliable evaluation of the trustworthiness of a product is extremely relevant to the users that have to decide whether to adopt the product or not. On the other hand, the knowledge comprised in an OSS trustworthiness model provides the developers of OSS with precious information about the qualities that they should guarantee.

In this paper we have presented the fundamentals of the QualiPSo model of OSS Trustworthiness. This model is different from the previous proposals in that it is both conceived to cover the issues that are typical of open source software, and it supports a balanced and complete evaluation of the software, addressing not only the technical characteristics, but also the economic, customer and growth/evolution issues that are often neglected.

Among the future activities that are planned in the context of the QualiPSo project, is the execution of the GQM plan described in Section 5. The resulting measures will be analysed in order to build a quantitative model that correlates the user perception of trustworthiness and the objectively measurable characteristics of the OSS. To this end, the “Q_user” quality focuses will be used as dependent variables of such model, while the “Q_actual” quality focuses will be used as the independent variables.

Acknowledgments

The research presented in this paper has been partially funded by the IST project  (<http://www.qualipso.eu/>), sponsored by the EU in the 6th FP (IST-034763); the FIRB project ARTDECO, sponsored by the Italian Ministry of Education and University; and the project “La qualità nello sviluppo software,” sponsored by the Università degli Studi dell’Insubria.

References

- [1] V. Basili and D. Rombach, “The TAME project: towards improvement-oriented software environments,” *IEEE Transactions on Software Engineering*, vol. 14, no. 6, 1988.
- [2] V. Basili, G. Caldiera, and D. Rombach, “Goal/Question/Metric Paradigm,” in *Encyclopedia of Software Engineering*, vol. 1, J. C. Marciniak, Ed.: John Wiley & Sons, 1994.
- [3] Morten Hertzum. The importance of trust in software engineers’ assessment and choice of information sources. *Information and Organization*, 12:1–18, 2002.
- [4] Larry Bernstein. Trustworthy software systems. *SIGSOFT Softw. Eng. Notes*, 30:4–5, 2005.
- [5] Alfonso Fuggetta. Open source software-an evaluation. *J. Syst. Softw.*, 66:77–90, 2003.
- [6] Peter G. Neumann. Robust non-proprietary software. IEEE Computer Society, 2000.
- [7] D. A. Wheeler. “Why open source software/free software (OSS/FS)? look at the numbers!”, April 2007.
- [8] ISO/IEC 9126-1:2001 “Software Engineering—Product Quality—Part 1: Quality model”, June 2001.
- [9] Boehm, Barry W., Brown, J. R, and Lipow, M.: Quantitative evaluation of software quality, International Conference on Software Engineering, 1976.
- [10] McCall, J. A., Richards, P. K., and Walters, G. F., “Factors in Software Quality”, *Nat’l Tech. Information Service*, Vol. 1, 2 and 3, 1977.
- [11] J. Smith David, D. Schuff, R. St. Louis, “Managing your total IT cost of ownership”, *Communications of the ACM*, Volume 45, n. 1, January 2002.
- [12] D. Taibi, L. Lavazza, S. Morasca, “OpenBQR: a framework for the assessment of OSS”, *Open Source Software 2007*, Limerick, June 2007.
- [13] R. Kaplan and D. Norton, *The Balanced Scorecard: Translating Strategy into Action*, Harvard Business School Press, September 1996.
- [14] del Bianco, V., Chinosi, M., Lavazza, L., Morasca, S., Taibi, D. How European software industry perceives OSS trustworthiness and what are the specific criteria to establish trust in OSS, QualiPSo report, October 2007, <http://www.qualipso.eu/sites/default/files/D5.1.1.pdf>
- [15] I. Ashley, “IT Balanced Scorecards – Suncorp’s Journey to a Contemporary Model”, *Australian Computer Society National Conference*, Melbourne, September 2004.
- [16] Luigi Lavazza, “Beyond Total Cost of Ownership: Applying Balanced Scorecards to Open-Source Software” *Int. Conf. on Software Engineering Advances*, Cap Esterel, August 2007.
- [17] “Making Open Source Ready for the Enterprise: The Open Source Maturity Model”, from “Succeeding with Open Source” by Bernard Golden, Addison-Wesley, 2005, available from <http://www.navicasoft.com>
- [18] Atos Origin, “Method for Qualification and Selection of Open Source software (QSOS), version 1.6”, April 2006, <http://www.qsos.org/download/qsos-1.6-en.pdf>
- [19] “Business Readiness Rating for Open Source - A Proposed Open Standard to Facilitate Assessment and Adoption of Open Source Software”, BRR 2005 - RFC 1, http://www.openbrr.org/wiki/images/d/da/BRR_whitepaper_2005RFC1.pdf.