Holistic (Software) Quality Theory. An Improved Definition and Meta-Model

Dobromir M. Dinev(✉)

New Bulgarian University, Sofia, Bulgaria
dobromir@gmail.com

Abstract. The current article introduces a short historical review of the roots of the holistic (software) quality theory. This review is used both as a presentation of the ground on which the holistic quality has been built upon and correspondingly to introduce a classification of the quality models known today. Since its introduction in 2017, the holistic (software) quality was several times enriched; one such advancement is part of the current work concerning the meta-model used to describe quality itself; thus, better control, management and etc., are achieved.

Keywords: Holistic quality · holistic quality definition · holistic quality meta-model · quality models classification · inherited quality · heuristic quality

1 Historical Roots of the Holistic Approach to (Software) Quality

The quality is deeply rooted in humankind's understanding of the world. Often when the question "What is quality?" is asked, it receives answers depending on the background of the person the question has been asked to. Among the answers, it can be noticed, clearly, the appearance of two words that signify the level of quality itself: "Good" and "Bad". If to be believed, the comprehension of "knowing the good and evil" [1] has been with us from the earliest days of our existence. If not to be believed, we should turn to more trivial things like food, a sense of security, art, etc.; then the first period in the formation of the term quality should be recognised, and its beginning should be referred to as this early moment when humans shaped those ground comprehensions and started distinguishing between the feeling of a secure environment versus the one that is not; furthermore, the situation where there is a "***good*** *food*" and "***bad*** *food*", the "*beautiful*" things verse that are not. Over the time, the humans' organisation became more sophisticated, and the quality started being recognised in the craftsmanship of particular individuals. This encloses the first major period of the term.

In the Roman Republic, those craftsmen gathered in small groups named collegia. The formation of the guilds boosted certain standardisation among the guild members and contributed further "better-quality" goods to be delivered. At the end of the 17th

© ICST Institute for Computer Sciences, Social Informatics and Telecommunications Engineering 2023
Published by Springer Nature Switzerland AG 2023. All Rights Reserved
T. Zlateva and G. Tuparov (Eds.): CSECS 2023, LNICST 514, pp. 134–143, 2023.
https://doi.org/10.1007/978-3-031-44668-9_10

century, the guilds start having the opposite effect on the quality by stopping innovation. Nevertheless, the guilds played their central role in the formation of the first universities centuries ago; for instance, the University of Bologna (established in 1088) [2], and this was leading new ideas to form. Such new ideas from the statistics and economics were at the base of the third period in the quality term to be present. It started with publishing a book that contains a definition of quality. The definition sees quality as an entity having two natures: objective and subjective. [3] The following decades were full of definitions of quality. By Crosby, "quality should be defined as "conformance to requirements" if we are to manage it". [4] Feigenbaum elaborates that the "Quality is customer determination […] it is based upon customers experience with the product and service, measured against his or her requirements - stated or unstated […] Product and service quality can be defined as: the total composite product and service characteristics of marketing, engineering, manufacture, through which the product and services in use will meet the expectations of the customer." [5].

The last of the quality fathers' definitions upon which the focus of the current work shall fall is about to be the definition of Juran. The definition has all the features from the previously reviewed. In addition, it has a direct referral to the presence of the defects and the "freedom" from them, stressing that "… it is most convenient to standardise on a short definition of the word quality as "fitness for use" [6].

Despite the first electronic computers being available in the first half of the 20th century, quality was not the focus then. Pieces of evidence that change are given from a conference held by NATO in 1968, where "about fifty experts from all areas concerned with software problems – computer manufacturers, universities, software houses, computer users, etc." [7, 8] gathered to debate topics concerning programming languages, software as a commodity, multiprogramming, time-sharing, and last but not least, the "problems of the large-systems" delivery. That last one relates to the quality and the price paid for achieving it. Such summits ignited a vast development in the quality field (see Fig. 1). In the second half of the 1970s, the first structural software product quality models start being present, for instance, the models of Boehm et al., [9] McCall's model. Richards and Walter [10] further recognised quality as a set of quality characteristics that are decomposable. A popular structural model created in the business was the FURPS model; [11] decomposed the quality into five major characteristics: functionality, usability, reliability, performance, and supportability. The structural software product models are advantageous when quality assurance engineers should model the quality of a particular product, especially when measurable quality goals are set. Nevertheless, the software branch was required to agree upon a standard that might have a broader possibility of being used by the software manufacturers. In December 1991, such an agreement became a fact with the actual issuance of the ISO9126 standard; it resembled many of the characteristics of the previously issued models. The authors of the standard recognise two levels of characteristics: major and sub-characteristics. Functionality, maintainability, usability, efficacy, reliability, and portability are recognised as major software product characteristics. ISO9126 standard is pivotal for the software branch and for the development of the software quality. The additions to the standard made in the following years give birth to the so-called compound models (ISO9126–1, much later ISO/IEC 25010), where not only the software product is in focus but a

wider view on the potential impactors (factors) for this quality to be embedded (despite the usability is an early recognised quality characteristic). Another positive impact was the equalisation regarding the terminology used to describe the software quality; furthermore, all this built the solid ground leading for the first attempts to build holistic (software) quality theory to be present, despite some of the broader quality ideas to be available in the first half of the 1980s.

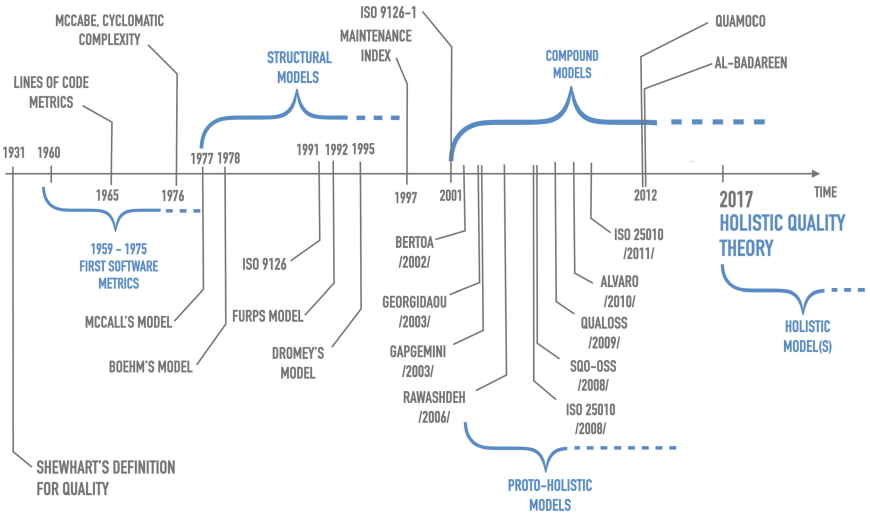

Fig. 1. Development of the software quality models.

At the beginning of the 2000s, a new software delivery business model emerged and became popular. The so-called "out-of-the-box software" or "off-the-shelf software" became an opposing idea to the bespoke. That new business model and technology advance boosted the development of a class of models based on ISO9126 that took some development and software usage specifics into account. The first model from that group was the Bertoa's quality model, which positions the following main characteristics: Functionality, Reliability, Usability, Efficiency, Maintainability, and Portability, but when it comes to the sub-characteristics, although most of them are directly taken from the ISO9126 standard, the sub-characteristics are divided into two major groups reflecting the "runtime" and the overall "lifecycle" of the software product. This is done to firmly settle the idea that the software product should be looked in a different perspective; thus, the software product quality is managed.

The author of the Generic, Multilayered and Customizable Model (or GEQUAMO) [15] was taking the business analysis perspective and fusion it with the quality assurance perspective, from one side and from the other, partially looking for a connection between those two, was introducing the viewpoint of operations as function in the company. The reason for the first relation is related to the apparent fact that each quality characteristic, which should be incorporated into the software product, first needs to be recognised and stated (primarily as a customer or end-user needs). A further important reason for the

suggested approach to quality is the managers, developers and end-users by utilising a framework (like the one suggested by the model) to build their own quality model and by assigning a weight to each quality attribute or characteristic of the relative importance of those attributes to be present and later taken into account when the software starts being created.

1.1 The Proto-Holistic (Software) Quality Models

In 1984, Garvin introduced his view on the matter of "*What is quality?*" and "*How is it possible to be defined*?" His understanding entails five possible approaches to defining the term quality. [12, 13] The first approach is the product approach; it looks like the structural product quality models, accumulating additional product characteristics like *durability* and *serviceability*. The *perceived quality* appeared under the product, despite the stand-alone major *user-based* perspective that is entirely related to the customer centricity visible in other quality definitions, for instance, in Crosby's definition. *The manufacturing approach* – gives the engineering perspective to the term quality. Garvin recognises it as the level at which the requirements are fulfilled. The *value-based approach* – adds the critical resource perspective, where time and money are considered. The last approach is the *transcendental approach.* The author believed that the term quality and the quality itself could not be defined precisely. There will always be a part of it that will be left undefined as it is impossible to be presented or represented by any given construct (like the tree structure of characteristics of the structural quality models). Publications like [14, 16] and some holistic quality theory scientific articles prove this statement mostly wrong. Notwithstanding, the model of Garvin should be considered as a proto-holistic quality model since it contains a much broader view of the term quality, not just the product perspective, the manufacturer's perspective, the lack of deficiencies as it is in the definitions of the quality father reviewed above, and it is not that generic as in the definitions of Shewhart and Zeithaml.

Rawashdeh's software product quality model is another instance of a model that focuses on non-bespoke software solutions. [16] The model also continues the tendency of the structural quality models; still, there is a gargantuan difference present on the first level; the model recognises a set of stakeholders at this first level: End-user, Analysts, Quality assurance, Business Owners, and Project manager; and have more than two levels. In the second level, each of those stakeholders' categories is connected with one or a few of the main quality characteristics. On the third and fourth levels, the model is organising the sub-characteristics of the product sub-characteristics and process sub-characteristics streams, positioning in each of them different sub-characteristics in each. Through the four steps, the model introduces a framework for creating customised quality models considering the different perspectives to achieve the accuracy of the quality model; the steps are *Step one*: identification of a small group of high-level quality attributes (characteristics) and utilising the "*devidire et concire*" method to subdivide them into sub-characteristics; *Step two*: distinguishing between external and internal metrics; *Step three*: identification of the user to each quality sub-characteristic (in terms of the model attributes; *Step four*: build the quality model, considering the initially introduced structure of the quality in the ISO9129 and Dormay's quality models.

According to [17, 18], the open-source models regarding the quality became in 2003; all of them are considering the alternative way of creating software opposing the commercial entities' created software. One more common feature can be easily identified across those models. They all thought of specific perspectives related to the open-source community, for instance, but not limited to maturity of the community, serviceability, etc., which is why they should be accepted in this group of quality models.

All proto-holistic (software) quality models share the same feature: they are broadening the views regarding quality. Still, none of them acknowledges and/ or reflects the holistic approach as a philosophy.

2 Classification in the Holistic Context of the (Software) Quality Models

Across the decades, different classifications with the main purpose of categorising the (software) quality models were suggested. The most sound classification related to the software branch is the one published in [19], known as the "Define, Assess, and Predict" classification (DAP). From a holistic point of view, the "Define, Assess, and Predict" classification is a necessary but not enough requirement for a model to be recognised as holistic. The missing additions, recognised for the moment, are from the next perspectives: processes (creation, transformation, etc.), the quality definition that includes the actors in the (software) product ecosystem, and one or few meta-models related to quality being present. This then excludes definitions like Shewhart's and Crosby's or standards such as ISO/IEC 25010. Following this ideology, the models in Fig. 1. (not only) are divided into the next few groups: structural product quality models (MaCall, Boehm, Dormey, OpenBRR, etc.), compound models (Bertoa, Alvaro, Gap Gemini, etc.), proto-holistic models (Garvin, Aaker, QualOSS, etc.) and holistic models (the holistic model suggested since 2017 by the holistic quality theory). To reflect certain settings and advancements in the development of software quality, the proto-holistic models are separated into two main generations: A typical model from generation one is such model that has a structural model that defines the term quality (like in ISO9126, later ISO25010) and there is some additional specific perspective which is considered an important; thus, the quality is achieved. For instance, in Rawashdeh's model, the recognition of the different stakeholders and the process; or in the open-source society models, the place of the open-source society itself is central for providing quality support of the delivered software product (SQO-OSS model).

The members of the second generation of proto-holistic models have in their list of characteristics a clear compliance with the DAP classification, specifically focusing on the prediction word in the abbreviation. Another significant difference between the two generations is the appearance of a meta-model in the second generation that helps to define quality as a term (QUAMOCO).

The holistic quality is compliant with all the characteristics mentioned by the proto-holistic group but also adds the previously mentioned perspectives. (Fig. 2.)

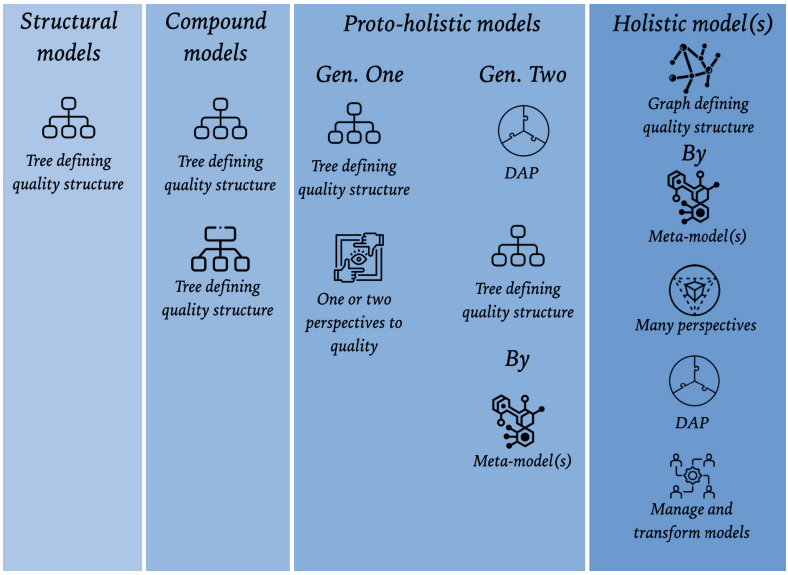

Fig. 2. Summary of the characteristics per category in the suggested classification.

3 The Holistic Definition of Software Quality and the Software Product Ecosystem

The definitions of Crosby and Garvin (reviewed above) introduced a set of stakeholders related to the software product quality; those were: *manufacturing*, the *customer*, *marketing*, and *engineering*. Such entities are recognised in the holistic quality theory as *impactors*.

The stakeholder management introduces a much bigger set of stakeholders, in comparison with the mentioned few, for instance, but not limited to Owners, Investors, Support, Finance, Sales, Government, NPOs/ NGOs, etc. (Fig. 3.) Each of those separate stakeholders can be assigned to a group. Each of those groups has an interest (in some cases legitimate) regarding the manufactured (software) product and the quality of it. This can be used for defining the quality reflecting the holistic philosophy.

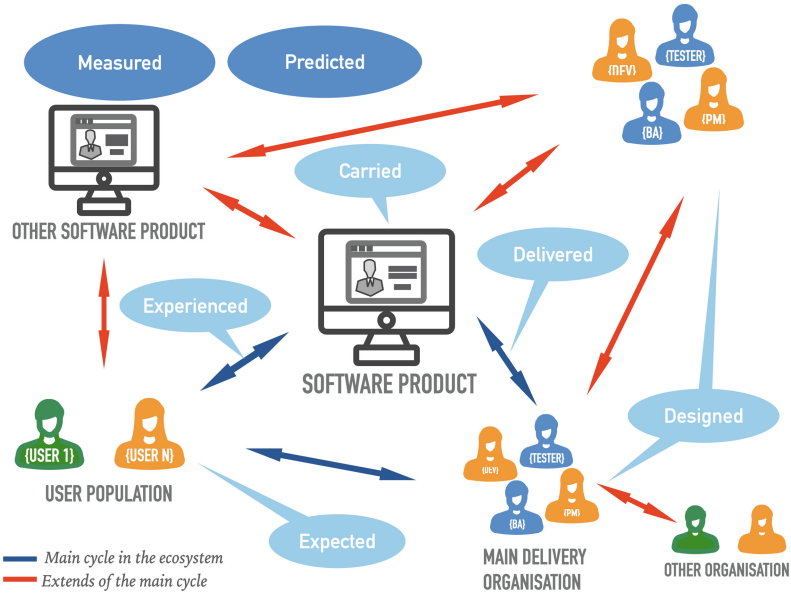

Fig. 3. Major groups of stakeholders in the software product ecosystem.

Utilising the "devidire et concire" principle and in clear desire for the quality later to be managed properly, the term quality is divided into the next seven sub-qualities; this was done reflecting Fig. 3. (Fig. 4):

Carried quality (CaQ) – the quality embedded in manufactured (software) products;

Expected quality (ExQ) – the expected by the users of the (software) product quality, usually in terms of quality components and levels for each component;

Experienced quality (ExpQ) – the quality actually experienced by the users during their use of the (software) product;

Designed quality (DesQ) – the quality that the creator of the (software) product intends to embed in the (software) product itself, i.e. the desired by the creator carried quality. This quality might be equal to none if the creator is not putting any effort into delivering quality;

Delivered quality (DeQ) – the quality that actually reaches the user of the (software) product;

Measured quality (MsQ) – the measured quality of CaQ, ExQ, ExpQ, DesQ, DeQ;

Predicted quality (PrQ) - the predicted levels of CaQ, ExQ, ExpQ, DesQ, DeQ, based on the MsQ.

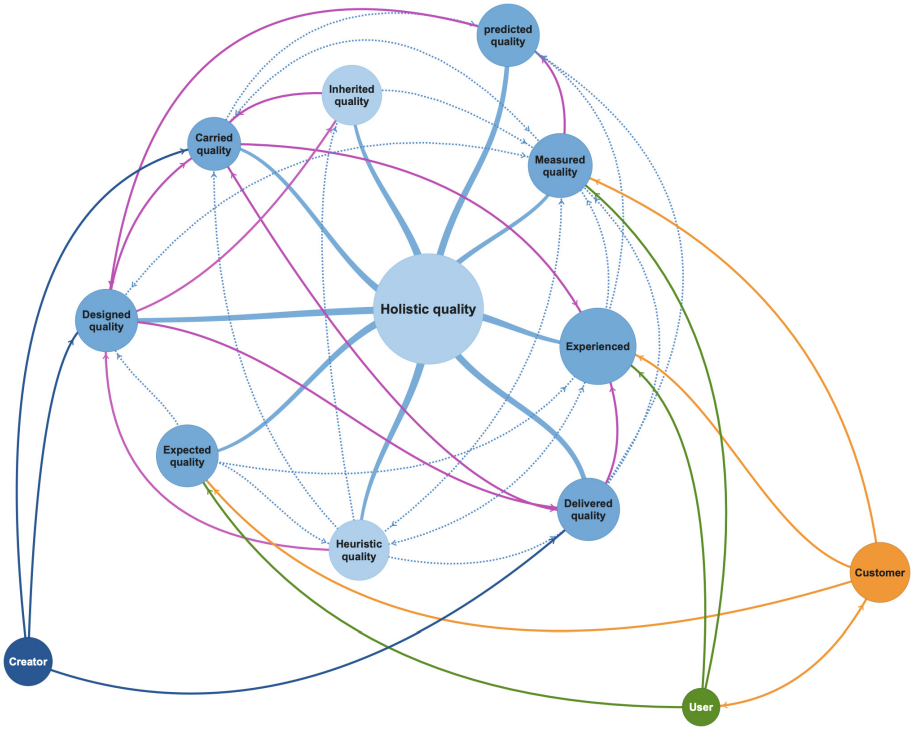

Fig. 4. Major groups in the software product ecosystem.

The additions are the Inherited quality (InhQ) – the quality that is "received" by using another (software) product in the delivery of the creator and Heuristic quality (HeQ) – the quality that is a result of the general rules (heuristics) taken from the use of the (software) product. An instance, from the software delivery branch, is a usability quality component-related heuristic regarding any given web sites stating: that the logo on the main page should not be a link to the same home page.

4 Updated Holistic Quality Meta-model

Following the acknowledgement of [20] for the need for an explicit meta-model to describe the increasingly complex structures of quality models, a meta-model for the software product was introduced. Figure 5 contains the new additions reflecting the management perspective around the quality components by which a proper reflection of the complete (software) product ecosystem can be done.

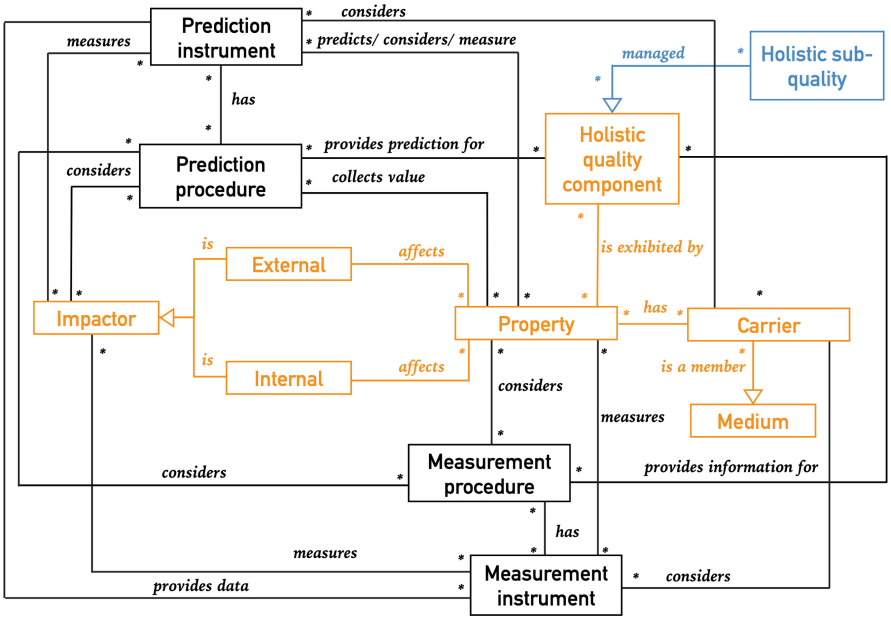

Fig. 5. Holistic (software) quality - updated meta-model.

5 Conclusion

The holistic approach to quality is relatively new happening on the (software) quality (assurance) landscape. The presented addition in the current article cannot be considered a major change to the previously stated theory. Still, the changes enrich the holistic approach making it even more complete. At the same time, it should be remarked that the holistic (software) quality is far from being finished, and the work will continue in the upcoming years.

References

1. The Bible, Genesis ph. 5
2. Rashdall, H.: The Universities of Europe in the Middle Ages, p. 150. Clarendon Press, Salerno, Bologna, Paris (1895)
3. Shewhart, W.A.: Economic Control of Quality of Manufactured Products (1931). Reprint p. 51 (2015)
4. Crosby, P.B.: Quality is Free, the Art of Making Quality Certain (1979). Mentor, ISBN: 978–0451621290
5. Feigenbaum, A.V.: Total Quality Control, p. e4 (2004). ISBN: 978–0070220034
6. M. J. Juran and Frank M. Gryna: Juran's quality control handbook, McGraw-Hill, e4,1988
7. Randell, B.: Software Engineering in 1968, Computing Laboratory – University of Newcastle upon Tyne (1979). p2, p6
8. Naur, P., Randell, B.: Bauer, F.L., Bolliet, L., Helms, H.J. (Eds.): Report on a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7–11 October 1968, p85, p87

9. Boehm, B.W., Brown, J.R., Kaspar, H., Lipow, M., Macleod, G.J., Merrit, M.J.: Characteristics of Software Quality. North-Holland (1978)
10. McCall, J.A., Richards, P.K., Walters, G.F.: Factors in Software Quality, National Technical Information Service (1977)
11. Grady, R.B., Caswell, D.L.: Software Metrics: Establishing a Company-Wide Program. Prentice Hall (1987)
12. Garvin, D.A.: What does "product quality" really mean? Sloan Manage. Rev. **25**, 25–43 (1984)
13. Stylidis, K., Wickman, C., Söderberg, R.: Defining perceived quality in the automotive industry: an engineering approach. Procedia CIRP **36**, 165–170 (2015)
14. Wagner, S., et al.: Operationalised product quality models and assessment: the Quamoco approach. Inf. Softw. Technol. **62**, 101–123 (2015)
15. Georgiadoui, E.: GEQUAMO-a generic, multilayered, customizable software quality model. Softw. Qual. J. **11**(4), 313–323. https://doi.org/10.1023/A:1025817312035
16. Rawashdeh, A., Bassem, M.: A new software quality model for evaluating COTS components. J. Comput. Sci. **2**(4), 373–381 (2006)
17. Adewole, A., Misra, S., Omoregbe, N.: A review of models for evaluating quality in open source software. In: International Conference on Electronic Engineering and Computer Science, IERI Procedia 4, pp. 88–92 (2013)
18. Haaland, K., Groven, A.K., Regnesentral, N., Glott, R., Tannenberg, A.: Free/Libre open source quality models-a comparison between two approaches. In: 4th, 2010 FLOS International Workshop on Free/Libre/Open Source Software, pp. 1–17 (2010)
19. Deissenboeck, F., Juergens, E., Lochmann, K., Wagner, S.: Software quality models: purposes, usage scenarios and requirements. In: Proceedings of the ICSE Workshop on Software Quality, pp. 9–14. IEEE (2009)
20. Kitchenham, B., Linkman, S.G., Pasquini, A., Nanni, V.: The SQUID approach to defining a quality model. Softw. Qual. J. **6**(3), 211–233 (1997). https://doi.org/10.1023/A:1018516103435