# Role, Importance and Significance of Software Quality

F. Témolé*, D. Atanasova**
* Capgemini Hamburg, Germany
** University of Ruse „Angel Kanchev“, Department of IIT, Ruse, Bulgaria
ftemole@t-online.de, datanasova@uni-ruse.bg

***Abstract* - In software development processes, as in other forms within the mechanical environment, the objectives of least costs, adherence to due dates and item quality must be taken into consideration. Great software system must satisfy different already concurred quality perspectives or product characteristics. Since of the gigantic significance of quality within the advancement of computer program items, its administration ought to be coordinates into software ventures. Software quality administration ought to make and keep up program quality with the assistance of specialized and hierarchical measures. As it were when the above-mentioned goals have been satisfied can one talk of financial computer program advancement. Software quality management can ensure that the software has the required level of quality. Quality objectives should be set, and their degree of fulfilment can be determined during development. In this paper, the essential findings from the expert interviews on the role and importance as well as the significance of software quality are brought together and a reflection is made between statements from business practice (experts) and science (theory), whereby the state of both research and practice can be decisively improved.**

***Keywords - component; formatting; style; styling; insert (key words)***

## I. INTRODUCTION

First of all, the term quality must be defined in order to subsequently be able to determine the term software quality. According to DIN EN ISO 9000, quality is understood to be a degree " to which a set of inherent characteristics of an object fulfils requirements" [3, 4]. Quality is thus reflected in how the product fulfils the existing requirements. There are numerous definitions in the literature that look at the term quality from different perspectives (e.g. objective or marketing-oriented view). Garvin [6] examined product quality, listed characteristic features and described five views (transcendent view, product-based view, user view, manufacturer view, value-based view).

TABLE I. FIVE VIEWS OF QUALITY, ACCORDING TO GARVIN [6]

| View | Description |
|---|---|
| **Transcendental view** | No precise quality definition possible, but is absolutely and independently recognizable from other quality characteristics as a whole. Only through the highest demands, standards and performance can this quality be achieved. |
| **Product-based view** | Quality can be derived from the product characteristics and is thus measurable via the properties or ingredients. |
| **User view** | Quality is characterised by the fact that individual customer needs are met and the customer is supplied with the most suitable product for his specific needs. |
| **Manufacturer's view** | Quality is achieved when given specifications or legal requirements are met. |
| **Value-based view** | Quality is characterised by the fact that the services are provided at a cost that is acceptable to the customer (price-performance ratio of the product). |

DIN EN ISO 9001 defines the requirements for a company's quality management and is designed to be industry-neutral. According to DIN EN ISO 9001 [4] a quality management concept must meet the following mini-mum requirements:

- Project-specific quality model with the relevant quality characteristics
- Determination of the parameters with the corresponding procedure
- Weighing the benefits and costs of the quality management system

Due to the special features of software products, a quality management adapted for software should be applied, which is to be regarded as part of the quality management of the company. Quality management can be carried out in a process- or product-oriented manner. According to Liggesmeyer [16] there are analytical and constructive methods that allow the development of comprehensible, reliable and easily modifiable software products and enable the early detection of errors. Software quality management can ensure that the software has the required level of quality. On the one hand, the quality of the software must be assured with appropriate measures (e.g. inspection of the requirements document), and on the other hand, the development process must be checked to ensure that defined quality characteristics are adhered to (e.g. regulations on the performance of tests). A distinction can be made between internal and external software quality.

## II. DIMENSIONS AND MEASURING QUALITY

### A. Dimensions of quality

Garvin [6] identified eight dimensions for analysing quality (Figure 1): The Performance dimension (1) includes the scope of performance, the primary operating characteristics of the product. The features dimension (2)

includes special or supplementary performance characteristics that go beyond the normal level. Reliability (3) reflects the likelihood that the product will behave consistently throughout its lifetime. The conformity dimension (4) indicates the extent to which the design, the expression of features corresponds to standards or previously defined specifications. The dimension durability (5) distinguishes between technical and economic durability. Technical durability is about durability until the product deteriorates to the point where repair is no longer possible. Economic durability refers to repair costs, as well as losses due to downtime. This means that the benefit of the product must be calculated up to the point where repair is no longer economical. The ease of maintenance dimension (6) refers to the speed, courtesy and competence of maintenance. The aesthetics dimension (7) assesses the external appearance and impression. Perceived quality (8) covers the way and feeling of the consumer in handling the product [6].


Figure 1. Eight dimensions of quality according to [6]

These dimensions of quality cannot simply be transferred to software products. Since there is no absolute quality, the determination of software quality is complex. Software quality can be circumscribed by numerous different properties. Quality models have therefore been developed for software products that contain quality criteria in the various dimensions. The defined quality characteristics can be used as a checklist to comprehensively ensure the quality requirements and thus provide a basis for estimating the resulting efforts and activities required during system development [9]

### *B. Internal and external software quality*

If the focus is on how well a system is built, then the internal software quality is meant, i.e. a consideration is made from the developer's point of view. Here it must be pointed out very early on that the software architecture also has a quality requirement, which is often omitted. This internal quality is not visible to the user. Important characteristics of internal software quality are, for example, readability, comprehensibility, extensibility, changeability, maintainability and reusability, all of which refer exclusively to the source code [19]. Internal quality ensures that the software can be better maintained in the long term and thus minimises the technical risks and costs as much as possible [17]. Therefore, it primarily addresses the technical debt and is an investment in the future for the respective system.

If the focus is placed on how the system behaves with regard to the specified expectations, then we are talking about external software quality, i.e. the focus is on the customer or user perspective. In this case, the quality is focused on the needs and requirements of the end user [15]. The technical correctness, reliability, ease of use, usability and performance of the system are important characteristics of external software quality. The higher the quality of the products we deliver, the fewer reworking afterwards. The big challenge is to identify the different needs of each user group and to meet their respective expectations [1].

### *C. Product and process quality*

Another view of software quality, as already explained, is product quality, which corresponds to the degree of quality of the product. This is often determined by how error-prone the system in question is. Its determination represents both a definable and measurable quantity and thus enables an objective assessment of quality. It is extremely important that we deliver functioning software. Software quality consists of determining the efficient and effective processes to ensure software development. A survey conducted by Javed and colleagues [12] showed that management plays a major role in quality assurance. It is the primary responsibility of team managers to support team members (e.g. through training) and provide them with a good working environment. Within the framework of quality management, measures of quality measurement, quality improvement and configuration management must also be provided [12]. Quality management in software development can be distinguished into

- Quality management focused on the software product and
- Quality management focused on software development.

Product quality is characterised, for example, by reusability, portability, reliability, manageability, flexibility and usability and depends on rigorous and systematic standards and procedures [18]. These quality characteristics determine the objective measurability of the quality of a software product [2]. Internal software quality, a characteristic of structural properties of the software product, includes the source code, the architecture and design of the software and the documentation (e.g. conception, requirements, API specification) [7]. Thus, only the requirements for the software product are considered, which can be used as criteria for product evaluation and which serve as guidelines for software development. Product-based quality assurance checks whether and how the developed software product fulfils these requirements. The disadvantage of this form of quality assurance is that errors are often discovered after the software product or

an intermediate product has been realised. Product quality can be achieved through testing and static quality assurance [19].
In order to improve software quality, the efficient and effective processes for software development need to be determined. In doing so, companies can pursue different strategies for software development [7]. Process-oriented quality assurance places requirements on the software development process that serve as guidelines for the design as well as the evaluation of the IT project. Through process-oriented quality assurance, software development is designed in such a way that as few errors as possible are committed. Various measures can be used to standardise and continuously improve software development in companies. Measuring the efficiency of the process, for example, is one of the most important items for process quality [11].

For the research design used, a sample size of 58 participants was chosen due to the relevance of different roles and stakeholders along the value chain, as well as the complexity of the software itself and the processes involved. . They have been interviewed within 4 months. The interviews were tape-recorded and subsequently transcribed. The expert interviews were scheduled for a minimum 90 minutes each. The researcher was not only an observer, but was also involved in the interview process as a moderator. Above all, their different views on software quality, the different hierarchy levels and their good and long-standing professional experience in the respective field of activity were taken into account. The focus here was on investigating several groups per subprocess at different points in time.

Some experts see an overall exposure to ever-increasing production and speed pressures. New requirements are not only being brought to developers more quickly, but are also becoming increasingly imprecise and are only clarified during implementation. This makes development and testing more difficult. Therefore, the work processes would have to be changed in order to keep up with the rapid changes, but then the entire quality assurance does not fit. The testing process and the quality process would have to be changed accordingly.
In the long run, the long development cycles will no longer be possible. Product launches must not take years as before, but must be faster. Frameworks, software, etc. need to be reviewed to enable faster feedback in the testing process. Test strategies (e.g. module testing instead of integration or regression testing) must also be reviewed.
It is therefore necessary that in the future there is no longer just reaction, but action and a stronger focus on automated and customer-oriented processes or products as added value for the customer. Accordingly, a central challenge will again be software quality, which will be characterised by even more frequent releases in the future. It is extremely important to have a quality model that takes into account the industry-specific characteristics and quality requirements. However, it should be adapted or reconsidered when the way of working in the company changes. It must be checked whether the quality model with its rigid stages can cope with the changes at all. Quality assurance must be adapted to the new structures.
Product-oriented and process-oriented quality assurance must not be considered independently of each other, as both are about software quality, but this is viewed from different perspectives. Both perspectives must therefore be taken into account.

### *D. Importance of software quality*

In companies, software quality is often only seen as a cost factor and the importance of software quality is not recognised. A good software must fulfil different, previously agreed quality aspects or product characteristics [21]. There is a high degree of agreement between literature and empirical studies on the importance of software quality per se. During the interviews it became clear that the importance of software quality is considered very high by most of the interviewees. Functioning software must be produced that is also used. The importance is rated or regarded as high to very high in the literature and by the experts. Quality is an important factor in the software industry [12]. However, some of the experts interviewed pointed out that software quality is overrated.
The assertion that the importance of software quality is not recognised was not confirmed either by the interviews conducted or by the literature. Only individual experts see software quality as overrated. However, it was found during the interviews that the significance of software quality is partly not known or not tangible. Many of the experts only considered product quality and hardly addressed process quality. Everyone understands software quality differently, so everyone has a different idea of how software quality is created. For example, the experts made the following statements about software quality:

- Good software quality is characterised by the fact that as few technical errors as possible occur.
- Software quality not only plays a role in software development, but is also related to documentation, test processes and test management.
- The software product manufacturer's ultimate goal is to deliver valuable and high-quality software to the customer that contains functionalities that the customer needs in a stable quality and has a high and stable performance. However, in doing so, there may be trade-offs in the functional scope.
- Software quality defines what the company delivers. Especially when the software is used productively, this also has a direct impact for the customer.

It became apparent that there is still uncertainty as to what constitutes software quality. The experts interviewed often only understand software quality as the

external software quality that meets the needs of the customer. Software quality is thus predominantly viewed from the customer's and user's point of view. According to the experts, one focus is on product quality during delivery and another, in terms of the quality dimensions, on performance, which, however, is not primarily in the foreground in the coordination process.

As elaborated from the literature, good software must also have internal software quality. The internal software quality can be captured by properties such as readability or changeability of the source code. This ensures better maintainability, for example. However, some software developers prefer to live out their freedom and individuality and do not recognise the necessity of rules and specifications for software development and thus for software quality. But quality also includes making decisions and not just doing what is required, but also questioning things from time to time.

Software quality should not be seen as a hindrance, but rather as an assistance. There is still a need for improvement in the company with regard to software quality. Often the company does not live up to it. The software developers must sometimes make their own decisions in order to improve the software quality. Not all software developers have the appropriate quality awareness. This has also been addressed in the literature. Javed and colleagues [12] pointed out that the attitude of the developer plays a major role in software quality. According to Distanont and colleagues [5] they are the main responsible for the quality of the software. However, developers often do not show cooperative behaviour or interest in the problem areas pointed out by quality management [5]. Therefore, the company's developers must be given a corresponding or an appropriate attitude towards software quality.

### *E. Causes of poor software quality*

In the literature and in the interviews conducted, several causes for a lack of software quality emerged. The most important are:

- High complexity in software development
- Rapid changes and adjustments
- Lack of knowledge and overview
- Scope of the desired functionalities
- Problems with requirements elicitation

The complexity of software development has increased considerably in recent years. Ever greater demands are being placed on software development. In addition, constant changes and adaptations (e.g. to customer requirements or legal regulations) are necessary, to which the software product manufacturer must react effectively and efficiently. This is also confirmed by the experts and the literature. Unrealistic deadlines and thus tight schedules are often agreed upon, which can be a major problem factor for software quality [12]. Quality is sometimes not sufficiently considered due to speed as well as efficiency.

Poor software quality sometimes also results from a lack of knowledge. There are fewer and fewer people who have an overview of the entire project. By breaking up into individual teams, knowledge is also decentralised. The global overview suffers due to too much focus on issues of the subject departments. As a result, errors are currently arriving in production that are caused by a lack of coordination between the developers. There are too many individual teams dealing with software quality. The interaction of individual sub-processes (e.g. requirements elicitation, software development and testing) is unbalanced and not very transparent. The global overview and the holistic view are missing. Literature and science agree on this. This can lead to product delays, rework and thus higher costs. Customer and employee satisfaction can decline. A large test coverage cannot close this gap; here too, there is agreement between literature and empirics. The author's statement was confirmed that software product manufacturers must implement the business and technically highly complex requirements in high quality in short development and change cycles while taking economic aspects into account.

Software quality must not only be seen from the perspective of the technicians, but must also take into account the perspective of the specialist/business departments. This is also the view of the departments that have to sell products or manage contracts. Functionalities must be made available to these departments efficiently and quickly in order to be able to hold one's own in the face of increased competition. Quality is therefore not an end in itself, but it must also not excessively delay or block development.

The software development of the software product manufacturer emphasises meeting the needs of different user groups. This was also evident in the responses of the different users of the software products. The software developers have to deliver a functional software, which means that functionality has a slightly higher priority. This is also the view of another interviewee. The goal in the development team is to create production-ready software more quickly, which means that quality takes a back seat. Other interviewees pointed out that it is better to make concessions in certain functional features, but that performance, stability, maintainability and security should be improved instead. Ergonomics, usability, but also maintenance and modularisation play a major role in the software development process alongside software quality.

As has already been worked out on the basis of the literature, errors in requirements elicitation cause enormous effort, longer development times and subsequent economic costs. Therefore, requirements engineering is absolutely necessary. Although this is carried out in practice, according to the experts it is not consistent and comprehensive enough, which leads to problems with the delivery of the software. The topic of software quality should already play a role in the requirements process and not only be examined afterwards during testing. Software quality therefore already begins with requirements elicitation in the business departments. However, most of the time the departments are not able to formulate the requirements

precisely, or they do not know the potential of the software products. This prevents correct technical implementation. Therefore, special attention should be paid to requirements elicitation. By clearly formulating the requirements, error chains or errors in software systems can be avoided [8]. This is also the view of the experts interviewed. However, there is usually not enough time for this due to time pressure. Only one expert stated that the current software quality in his division is very good, so that other challenges (e.g. timely elicitation of requirements) are currently in focus.

The multitude of quality characteristics allow the existing expectations and needs of different experts along the software development to be taken into account. When viewed from the perspective of the developer and the software architect, for example, the focus is on how well a system is built technologically. This aspect of quality is not visible to the user. Important characteristics of this view include readability, comprehensibility, extensibility, changeability, maintainability and reusability, all of which relate exclusively to the source code. It also serves to make it easier to maintain the software in the long term and thus to minimise the technical risks as much as possible.

### *F. Value of software quality*

Software quality also pays off financially for companies, especially for highly complex, maintenance-intensive systems [22]. The experts were therefore asked how the value of software quality is assessed in the company. Some experts said: With software quality that is taken into account right from the start, fewer corrections have to be made afterwards, so costs can be saved. The higher the software quality, the fewer problems and errors occur and the less rework the software development and IT staff have to do.

However, the economic benefits are usually not clear to the interviewees. The experts confirmed the author's statement that savings are often made in software quality assurance measures due to permanent cost pressure. The economic perspective of software quality must be communicated more intensively to those involved. Through the interviews and observations conducted, it became apparent that mechanisms for error prevention have not been established or sufficiently taken into account in all areas of the company. This leads to problems in the timely detection of errors.

### *G. Importance in the company*

Based on the literature, it was worked out that software quality must be integrated as an integral part of the corporate strategy. The experts were therefore asked about the importance of software quality in the company. One interviewee stated that efficiency and speed are more important in his division than ensuring quality. Market entry and thus presence are paramount. In another division, software quality also plays a rather subordinate role from the company's point of view. It is often only considered as a partial aspect in retrospect, so that quality assurance does not take place from start to finish, but only when certain incidents have occurred. However, many of the experts interviewed saw a causal connection between corporate success and high product-based software quality. As a result, expensive rework could be reduced. One interviewee stated that it is part of the company strategy to deliver high-quality software.

The image of quality assurance has changed in recent years. Overall, the company has seen a significant improvement in the quality of software. However, software quality and testing are often neglected due to time pressure or in order to save costs. In practice, software quality often has to be weighed against quality. A better error culture must be lived in the company. In the literature, company policy is also identified as an important aspect of poor software quality [12]. Within the company, specific efforts must be made to ensure that all those involved in software development recognise the importance of software quality within the company. The literature has shown that there is a causal relationship between the pursuit of high product-based software quality and the company's success. However, this is not yet practised by all divisions of the software product manufacturer.

Sufficiently high software quality allows systems to be developed more economically. Only high-quality software will be used in the long term and ensures good maintainability. Existing systems could be further developed and new systems introduced with less effort, mainly in terms of personnel, if more attention were paid to the issue of quality.

## III. CONCLUSION

As the interviews showed, the experts in the company tend to have a product-based view of quality, which is expressed purely through the design of the software itself. It is also clear from the expert interviews that a view-based model - such as the quality model according to IOS/IEC 25010 - is better suited for a holistic control of quality, since here the needs of the various user groups are the focus of consideration.

With regard to software quality, therefore, it is not the favoured user view that is in the foreground in the company (despite the agreement between theory and practice) [13], but the manufacturer's view in combination with the aforementioned product-based view. Thus, in the company, quality is considered to have been achieved when the product fulfils the functionalities and thus the given specifications or design requirements have been met [15]. All in all, this is to be seen rather negatively with regard to user orientation, improvement of software quality and securing the economic future of the company. Thus, in comparison with the literature, it can be formulated that the role, importance and significance of software quality [14, 20] in terms of its effect on the success of the company is not sufficiently taken into account.

The economic advantages of software quality are usually not clear to the interviewees or, due to cost and time pressure, savings are often made in software quality assurance measures. The value of software quality is partly recognised, but not consistently implemented in the

company. Software quality is often only seen as a cost factor and savings are made at this point.

The most important approaches to reviewing software quality are improvement, delivery time, overheads, the inclusion of customer requirements and how quickly the customer's requirements are implemented. Software quality management provides a framework for software development by defining and measuring software quality in the first place. The success factor software quality is largely dependent on how well the respective company succeeds in integrating and consistently using it as a fixed component of software development.

## REFERENCES


[1] ASQF. (2016). Quality from the start: 20 years in the service of quality. (ASQF, ed.) Potsdam.

[2] Broy, M., & Kuhrmann, M. (2021). *Introduction to software engineering.* Berlin: Springer.

[3] DIN EN ISO 9000. (2015). Quality management systems - Fundamentals and terms. Berlin.

[4] DIN EN ISO 9001. (2015). DIN EN ISO 9001 Quality management systems - Requirements.

[5] Distanont, A. (2015). The Test Automation Adoption-A Case Study. In: Numprasertchai, H. & Meeampol, S. (Eds.): International Journal of Business Development and Research. Volume 2, Issue 1, pp. 60-78.

[6] Garvin, D. A. (1984). What Does "Product Quality" Really Mean? *MIT Sloan Management Review, 26*(1), pp. 25-43.

[7] Hassan, M., Mubashi, M., Shabi, M., & Ullah, M. (2018). Software quality assurance techniques: A review. *In: International Journal of Information, Business and Management, Vol. 10, No.4, 2018*, pp. 214-221.

[8] Hoffmann, D. (2008). *Software Quality.* Berlin Heidelberg: Springer.

[9] ISO/IEC 25010. (2011). "Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models". ISO/IEC. Retrieved 18 March 2017 from iso.org.

[10] ISO/IEC 9126-1. (2001). Information technology-Software product quality. Geneva.

[11] Jamil, M., Arif, M., Abubakar, N., & Ahmad, A. (2016). Software Testing Techniques: A Literature Review. *In: 2016 6th International Conference on Information and Communication Technology for The Muslim World*, pp. 177-182.

[12] Javed, A., Maqsoo, M., Quazi, K., & Shah, K. (2012). How to improve software quality assurance in devoloping countries. *In: Advanced Computing: An International Journal ( ACIJ ), Vol.3, No.2, March 2012*, pp. 17 - 28.

[13] Kitchenham, B., & Pfleeger, S. L.. (1996). Software quality. *IEEE Software, 13*(1), pp. 12-21.

[14] Kreilkamp, E. (1987). Strategic management and marketing: market and competitive analysis. Berlin: de Gruyter.

[15] Lange, C. (2011). *Software quality models.* Munich: TUM.

[16] Liggesmeyer, P. (2009). *Software Quality.* Heidelberg: Spektrum Akademischer Verlag.

[17] Petrasch, R. (2001). Introduction to Software Quality Management. Berlin: Logos Verlag.

[18] Rashid, J., & Nisar, W. (2016). How to Improve a Software Quality Assurance In Software Development - A Survey. *In: International Journal of Computer Science and Information Security (IJCSIS), Vol. 14, No. 8, August 2016*, pp. 99-108.

[19] Spillner, A., Roßner, T., Winter, M., & Linz, T. (2014). *Praxiswissen Softwaretest: Testmanagement* (4th ed.). Heidelberg: dpunkt.Verlag.

[20] Venelinova N., D. Antonova and I. Kostadinova (2021). Adaptive Approach of System-engineering Project Management Skills Acquisition MIPRO 2021 - The 44th International Convention on Information, Communication and Electronic Technology, Proceedings MIPRO2021.Engineering Education, , 1778-1784, doi:

[21] Wallmüller, E. (2001). "Software Quality Management in Practice: Software Quality through Leadership and Improvement of Software Processes" (Vol. 2). Munich Vienna: Carl Hanser Verlag.

[22] Zimmermann, T. (05 March 2018). *Software quality*. From testing costs - not testing too!: https://www.business-wissen.de/artikel/software-qualitaet-testen-kostet-nicht-zu-testen-auch/ retrieved