Software Quality Assurance

ANSHUL GUPTA [1*], JUHI GAUTAM [2], NEHA GUPTA [3]

[1*, 2 & 3] *Dronacharya College of Engineering, Farrukhnagar, Gurgaon ,*
coolanshul2704@gmail.com [1], gautam.juhi3@gmail.com [2], guptaneha234@gmail.com [3]*

Abstract

This paper describes the author's experiences, gained during a complicated and long outsourcing project. The applied Software Quality Program is presented and possible approaches to these problems, ideas for its improvement are given. The problems of project management, software configuration management and maintenance are clarified. An achievable-both incremental and measurable - has been suggested. This paper sets out a number of challenges faced by software quality community. These challenges relate to the outspread view of quality and the consequences for software quality definitions. These definitions are related to eight perspectives of software quality in an end-to-end product life cycle. In conclusion the author shares a few possible directions for further research.

Keywords: Software quality assurance, Out sourcing,WWW

1. Introduction

Outsourcing means that software engineering activities are contracted to a third party who does the work for lower cost and higher quality (hopefully). Due to the shortage of software people with appropriate qualification, skills and experiences, or lack of other facilities (hardware, development environment, communications, etc.,) the contractor looks for a vendor to solve these problems. Outsourcing can be imperfect– when only a few parts of the software system are contracted, or complete, when the whole software system under development is contracted. According to another classification the outsourcing can be planned or ad-hoc. The planned outsourcing is a part of company imperative business plan to beat new markets, to enter the market very fast or to create an attractive image, not having the resources needed. The ad-hoc outsourcing can be an attempt to deal with a project behind schedule or to solve unexpected software problems (McCall et al.,1997).

From the other side the benefits for the vendor are to obtain a real and pure software work keeping his/her people busy and comparatively well paid. This can be a way to avoid some software company pre-project activities and a number of post-project activities (Boëhm,1978).

Research relating to software quality is typically rooted in the study of product quality factors and the usability of those products in a context. During this research and study, emphasis on quality assurance and measurement is limited to this product perspective. Furthermore, the domain in which quality is measured is limited to that of Information Systems (IS). As far as it relates to the IS domain, the paper first considers definitions of quality and other related issues.

As evidenced by the needs of eCommerce it is also necessary to develop the study of software quality to embrace other domains like the World Wide Web (WWW). In this domain, product quality necessitates the study of additional quality factors, which addresses access, interaction and navigation. Furthermore, the owners of eCommerce solutions have new expectations that they will gain competitive advantage from their sites and this introduces further perspectives of software quality beyond that of product quality.

2. A Case Study - our Outsourcing Project

2.1 The Project

In the early 1990's a number of sequential software development projects were carried out in Bulgaria, ordered by the software development department of a big German corporation. All projects were in the field of software development and other types of dedicated software. The purpose was to develop a software tool for the analysis, planning and design of computer-based training programs. As a member of the development team,

I would like to share our experience gained during this complex and long outsourcing project. Further, in this paper we will call the representatives of the German company, the initiator of the project, with a general name customer, and the Bulgarian team – developer (Crosby, 1984) .

Bearing in mind the well known from the literature software engineering recommendations, both sides made an attempt to define explicitly the initial conditions, constraints and risk factors and to outline the basic principles of project management and quality assurance program.

2.1.1 The following constraints were imposed on the Bulgarian developers:

- The development cost and the project deadline had to be determined in advance and could not be reconsidered;
- The general characteristics of the hardware platform and the software tools for the project had to be determined by the customer, but
- The selection and purchase had to be made by the developer;
- The developer had to articulate and submit for approval the project management scheme and the quality assurance plan;
- During the development the customer had to be responsible for the requirements definition, external specifications and the independent testing, and
- The developer had to be responsible for the detailed design, programming, internal testing and preparation of the maintenance documentation.

2.1.2 At the beginning of the project some risk factors have been identified and analyzed, namely:

- The requirements definition was incomplete and fuzzy, not allowing the precise estimation of the size and complexity of the desire

2.1.3 Software system:

- The distant development was to some extent new for all participants and the lack of experience required finding out the appropriate

2.1.4 Solutions by testing several alter-natives:

- In this distant development the daily communications by e-mail should be in a language (English) foreign to both sides.

2.1.5 The SQA program has been created on the following three assumptions:

- The quality of the software product depends on the quality of the people involved in the project and their proper management
- The quality of the software product depends on the quality of the development process;
- The quality of the software product in a remote development project depends heavily on the quality of the communications system.

The Project Management program described the gradual development, the responsibilities of the teams and a detailed schedule for all distant activities and face-to-face joint meetings.

2.2 Project results and their evaluation

The work on the developed tool had several continuations and finally ended (due to some financial reasons) a year ago. Now from the distance of the time passed we can evaluate the project and to summarize the results of this experimental outsourced software development.

We can report a successful Software Quality Assurance (SQA) program. The de-fined set of three assumptions really managed to ensure an efficient SQA policy. Next we will remind each assumption and will describe our brief comment for it.

2.2.1 Assumption 1.

The quality of the software product depends on the quality of the people involved in the project and their proper management.

The size and the structure of the teams, the requirements for education, qualification and experience of the developers, the scope of their work and responsibility have been planned carefully and followed strictly. The high professional level gave us the possibility to solve all arising problems (both algorithmic and technical). The developers were motivated not only by their salaries but by the challenges of the project itself. A number of research papers, describing some problems and their innovative solutions have been published during and after the project

The lesson: Select the right partner to be qualified, loyal and highly motivated.

2.2.2 Assumption 2.

The quality of the software product depends on the quality of the development process.

A model for a step-by-step development has been implemented. Two consecutive prototypes have been created. Unfortunately, the preliminary defined schedule was modified several times due to very often changes of the requirements.

The lesson: It is not possible to predict the duration and cost of each project phase. So as a step-wise procedure for project scheduling should be applied: a rough schedule with definition of the basic phases at the beginning of the project and a refinement only for the next phase, specifying the expected results, the acceptance criteria and the proportional payment.

2.2.3 Assumption 3.

The quality of the software product in a remote development project depends heavily on the quality of the communications system.

The whole organization of the communication (who, when, how often, the control and distribution rules, etc.) was described in the so called communication protocol. Those days when the Internet made its first steps we encountered many technical problems. Fortunately they are not valid anymore because of the reliable and fast Internet connections now.

The lesson: A communication protocol has to be established at the beginning of the project and has to be strictly followed.

3. A modify approach to outsourced software development

The experience from a set of distant projects and the additional research work give us the possibility to propose a number of improvements to the overall development style in software outsourcing.

3.1 Our approach to SE activities

We describe our general approach to the software engineering activities, performed in a software company. Here we will present its basic principles only to make possible for the reader to understand how they can be applied to outsourced development.

The basic characteristics of our approach are the following:

- The approach is an incremental. At the beginning it can be applied only for one software engineering activity – e.g. Software Quality Assurance. After the evaluation at the end of the period for experimental use, the approach can be expanded to cover other activities, too.
- The approach is based on the idea to keep the control over the whole project all the time. But it is well known that we cannot control what we cannot measure. So we propose to use the universal technique of comparative analysis and a set of supporting tools so as to measure the progress in any observed activity;
- The approach is adjustable to the specific features and style of work in a given soft- ware company. For each applied technique at least two different levels of complexity should be defined. After the corresponding cost/benefits analysis the senior managers of the company can decide which activi-

ties will be covered and to what extent. In our opinion this makes the approach more pragmatic and feasible.

The introduction of this approach to given software company comprises the following steps:

- » Defining the main goal to be achieved (e.g. an efficient development process, high quality software products, stable and attractive market image, etc.)
- » Selecting a few basic SE activities to start with;
- » For any activity - developing a systematic methodology and a plan for its implementation;
- » Starting a pilot project to examine the validity of the concrete approach version.
- » Evaluating of the results and making a decision how to proceed.

Table .1 The outline of the basic activities content (Eskenazi , Maneva & Radev 2000; http:// www.testlabs.com/ white_papers.html)

Activity	Method/Technique	Comment
Pre-contract and Contract Activities		
Select the software parts to be out- sourced	Cost/benefits analysis	
Choose the right developer	Comparative analysis	Criteria: qualification, experience in shipped products, standard and stable development environment, QA practice
Create a con- tract and assess it	Formal review through checklists	To achieve the SQA goals state a number of milestones with defined deliverables, acceptance procedures and payments
Management		
Personnel management	People management capability maturity model	A number of key practice areas for software people should be identified
Product management	Comparative analysis to measure the progress	Criteria: size, complexity, functionality, flexibility, etc.
Process management	Incremental development through consecutive proto- types	Well specified local and umbrella activities
Project management	A rough schedule for the whole project and a detailed schedule only for the current and the next stage	Project tracking, risk analysis and mitigation
Software Quality Assurance		
Testing	Static and dynamic methods	Unit testing at the developer's site and complete and repeatable system test at customer's environment
Quality evaluation	Quality evaluation models and methods	Software ranking for a set of quality Characteristics
Quality control	Reviews, software metrics, comparative analysis	Create a SQA plan and follow it
Software Configuration Management		
Change con- Troll	Regulations for changing any item, "freeze" requirements	Auditable and repeatable creation of soft- ware using a controlled project library
Configuration control	Configuration audit for each increment	Release notes - to restrict the dependency on the developer

3.2 A program for an efficient outsourcing development

We would like to apply our approach to the outsourcing development. Following the above mentioned steps, first we have to define the main goal of the program. The goal is to keep the control over the people, product, process and project. This goal leads to the initial set of three activities– two with already recognized importance- "Project Management and Quality Assurance" - and a new one, underestimate till now – "the Software Configuration Management". We decide to include it because the analysis of project results shows the need of special efforts for change management. After the final acceptance test of the software system the initial set of activities should be expanded with the maintenance activity. The outline of the basic activities content is given in the Table 1.

3.3 Maintenance of the Outsourced Software Products

The problems of maintenance after the end of the outsourcing project are not well studied till now. In our project after the 12-months guarantee period for corrective maintenance there was a number of additional contracts for adapting and enhancement. Generally speaking, three approaches are possible:

- » Maintenance is done within the developer's organization. This was our case. It seems to work for a complete outsourcing and can be even cost-efficient because no need for additional efforts to understand the programs so as to change them. The main is advantage was that many members of the developer's team had already other primary responsibilities and couldn't devote much time to this "old" project;
- » Maintenance is done within the customer's organization. It seems to work for both partial and complete outsourcing projects, but it is possible only in case there is a permanent group, working together with the developers especially to be prepared for the future maintenance;
- » Maintenance is contracted to a third part. This situation is a imagined – who will dare (and for how much money?) to maintain software, created by others?

We believe that after a successful SQA and Project Management program accomplishment the maintenance efforts will be pretty small, but anyway we need prescriptions for them. We start an investigation on this topic and we hope that our general approach will work for maintenance, too.

4. Quality defined

There are many different definitions of quality. It is typically defined in terms of conformance to specification and fitness for purpose. Figure 1 shows a number of acknowledged definitions

There are difficulties with definitions that focus on conformance to specification and fitness for purpose. In the first instance it follows that if there is a deficiency in the specification then there will be a deficiency in the quality, yet the definition would imply that conformance to the specification will produce a quality product. This is not the case and an inferior or deficient specification will not produce a high quality product. Fitness for purpose can also be challenged along the same lines. For example, there are many types of motor cars that are fit for the purpose of transportation of two to four individuals from A to B. But they are not all Rolls-Royce quality cars. So, fitness for purpose does not fully define quality either. International standards organization also define quality (ISO/IEC 9126-1, 2001).

When the quality relates to software quality it is mainly defined in terms of characteristics of a product and its use. There are two very important points in these definitions. The first is, they emphasise the product and in the case of software this is the application delivered to the purchaser. The second is that they introduce the desirability of measurement by using words like totality and degree

This is in keeping with a natural description of high or low quality which in scientific terms might equate to a scale such as 0 to 100. In the domain of Information Systems, quality is limited to measuring the attributes (the quality factors) of the software product and measuring its use. This is the narrow view of quality which only addresses quality-of-process, quality-of-product and quality-of-use.

A broader view of quality is suggested by the founding father of the Japanese quality movement, Kaoru

Ishikawa. His view of quality is shown in Fig. 3.

On this basis, it follows that limiting software quality to the process by which the product is built and to its usability is too narrow a view and that there are a number of perspectives of quality (some of which are not widely researched). Eight perspectives are represented on the newly extended Software Quality Star mark II as illustrated in Fig.4.

The Software Quality Star mark II is an enhanced version of the original model. Its original motivation was to illustrate the principal points of focus in ISO 12207 which relates to software life cycle processes. Mark II is enhanced to incorporate end-to-end perspectives together with domains like the World Wide Web which are additional to the Management Information Systems domain.

The eight perspectives in the Software Quality Star are quality-of-procurement, quality- of-contract, quality-of-production, quality-of- project, quality-of-process, quality-of-product, quality-of-use and quality-of-maintenance. So, it is appropriate to step back and consider quality on a higher level. It can easily be argued from the definitions in Figures 1, 2 & 3 that quality is a measure of something (other than product characteristics) relating to the different perspectives and this paper proposes that at the higher level quality is a measure f excellence. The excellence should then be quantified for each perspective. For example, in the case of quality-of-product, the excellence will relate to product external and internal quality factors. In the quality-of-production perspective the excellence will relate to the producer considerations and in the perspective of the owner the excellence relates to procurement and issues like value for money and competitive advantage. So, software quality could be defined in terms of a measure of excellence in the perspectives of the end-to-end software product life cycle.

4.1 Quality in life cycle models

Fig.1. *Definitions of quality (Crosb, 1984; Deming, 2000; Feigenbaum, 1961; Juran, 1989; Oakland, 1993; Shingo 1988; Taguchi,1987)*

Acknowledged authority	Definition or interpretation of quality
Crosby, 1984, p80	Conformance to requirements.
Deming, 2000, p168/9	Quality can be defined only in terms of the agent. Who is the judge of quality? Deming continues, "*The problems inherent in attempts to define quality of a product... were stated by the master Walter A Shewart (1986, Ch 4) viz the difficulty in defining quality is to translate future needs of the user into measurable characteristics, so that the product can be designed and turned out to give satisfaction at a price that the user will pay*".
Feigenbaum, 1961, p13	The composite product characteristics of engineering and manufacture that determine the degree to which the product in use will meet the expectations of the customer.
Juran, 1988, p11 1989, p15	Quality is product performance, quality is freedom from defects, quality is fitness for use. Fitness for use
Oakland, 1993, p4	Meeting customer's requirements.
Shingo, 1988, p11	Zero defects.
Taguchi, 1987	Product quality is determined by the economic loss imposed upon society from the time a product is released for shipment.

The third challenge addressed in this paper focuses on quality in the life cycle. Popular conceptual system-life-cycle models are software engineering focused with processes mainly centered on the creation of the software product. That is, they address quality- of-process, quality-of-product and quality-of-use in a context of use. But the broader view of quality dictates that a life cycle that focuses only on software development is insufficient and that a full end-to-end software product life cycle is required. Such a model would embrace

quality from product conception through to product retirement and would address all of the quality perspectives of the Software Quality Star – Fig.4. That is, quality-of-procurement, quality-of-contract, quality-of-production, quality-of-project, quality-of-process, quality-of- product, quality-of-use and quality-of- maintenance.

Fig.2. *International standards definitions of quality (ISO/IEC 12207, 1995; ISO/IEC 9126-1)*

Standards Body	Definition of quality
German Industry Standard DIN 55350 Part 11	Quality comprises all characteristics and significant features of a product or an activity which relate to the satisfying of given requirements.
ANSI Standard (ANSI/ASQC A3/1978)	Quality is the totality of features and characteristics of a product or a service that bears on its ability to satisfy the given needs.
BS 4778,1987 (ISO 8402, 1986)	The totality of characteristics of a product or service that bear on its ability to satisfy stated or implied needs.
IEEE Standard (IEEE Std 729-1983)	**a** The totality of features and characteristics of a software product that bear on its ability to satisfy given needs: for example, conform to specifications. **b** The degree to which software possesses a desired combination of attributes. **c** The degree to which a customer or user perceives that software meets his or her composite expectations. **d** The composite characteristics of software that determine the degree to which the software in use will meet the expectations of the customer.
ISO/IEC 9126 (1991)	The totality of features and characteristics of a software product that bear on its ability to satisfy stated or implied needs.

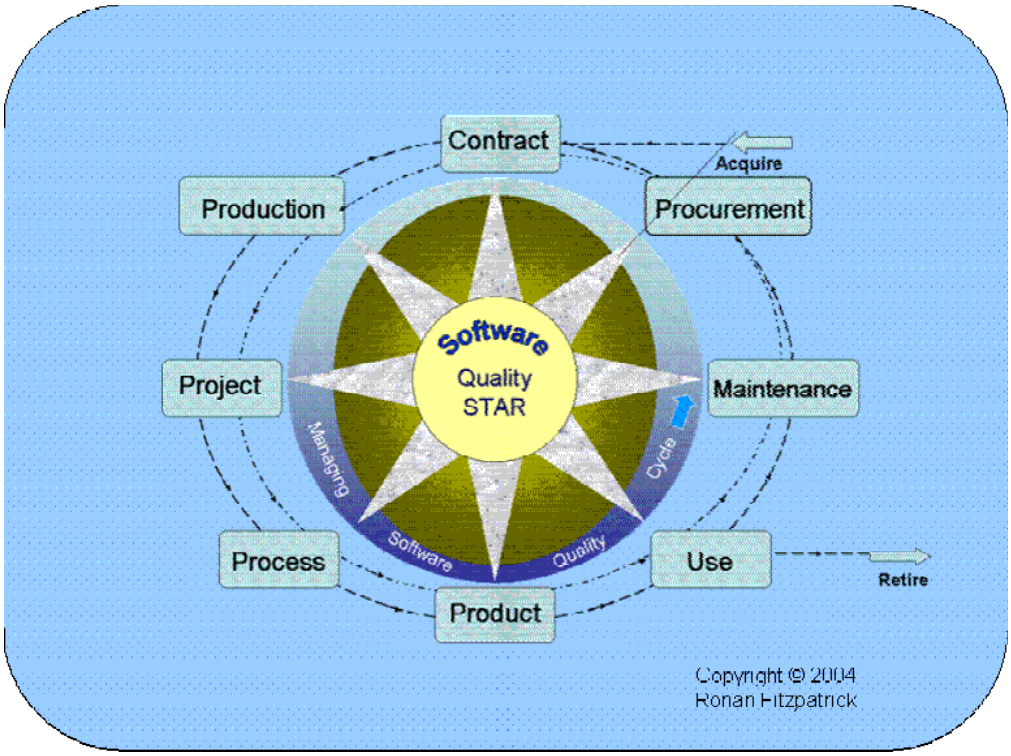
Fig.3. *Ishikawa's broader view of quality (Kaoru Ishikawa, 1985)*

Ishikawa 1985 p44/5	In a series of definitions relating to quality control he refers to products which can '*satisfy the requirements of consumers*' This he explains should be "*Narrowly interpreted to mean quality of products*". He continues "*that Broadly interpreted quality means quality of work, quality of service, quality of information, quality of process, quality of division quality of people including workers, engineers, managers and executives, quality of system, quality of company, quality of objects etc. To control quality in its every manifestation is our basic approach*".

This end-to-end model also addresses the fact that the word quality is not mentioned in the popular conceptual system-life-cycle models. Expressions like validation and verification or test or evaluation are used but quality as a focus of management during the life cycle is not given the significance it merits. This contrasts with the inclusion of risk management in Boëhm's spiral model. The traditional approach to quality relates to the term Quality Assurance (QA) which is associated with code testing. It is more appropriate to refer to managing software quality in order to emphasize the on-going end- to-end life cycle aspects. This,

too, is illustrated in the Software Quality Star mark II by the cyclical-flow dotted line shown in Figure 4

Fig.4. *Software Quality Star mark II (SQ-Star) (Fitzpatrick , 2003)*



Having identified the eight different quality perspectives (Section 2) it follows that each perspective has its own interpretation of quality. For example, when interpreting quality-of- product perspective, the topics of interest are product quality factors. Likewise, when interpreting the quality-of-procurement (ownership) and quality-of-production the topics have to do with procurement factors and production factors. Such a set of factors was identified as software quality strategic drivers. These are presented in Figure 5 where they also include familiar software quality terminology for each driver.

5. Redefining quality for evolving technologies

As part of quality-of-use, ISO 9126-1 explains the need to refer to context of use, that is, one product with opportunities to use it in different contexts. While the context of use may change, the domain of use is consistent - the domain is Information Systems. But the World Wide Web (WWW) is a different domain and different quality factors apply.

Multiple domains (typically the WWW) are illustrated in Fig. 4 & 5 by the second cyclical-flow dotted line. Five additional quality factors for the WWW were identified. These five are visibility, intelligibility, credibility, engagibility and differentiation and are shown together with their sub- characteristics in Fig. 6.

Fig.5. *Producer and Procurer strategic drivers of software quality (Fitzpatrick 2001, Fitzpatrick & O'Shea, 2004)*

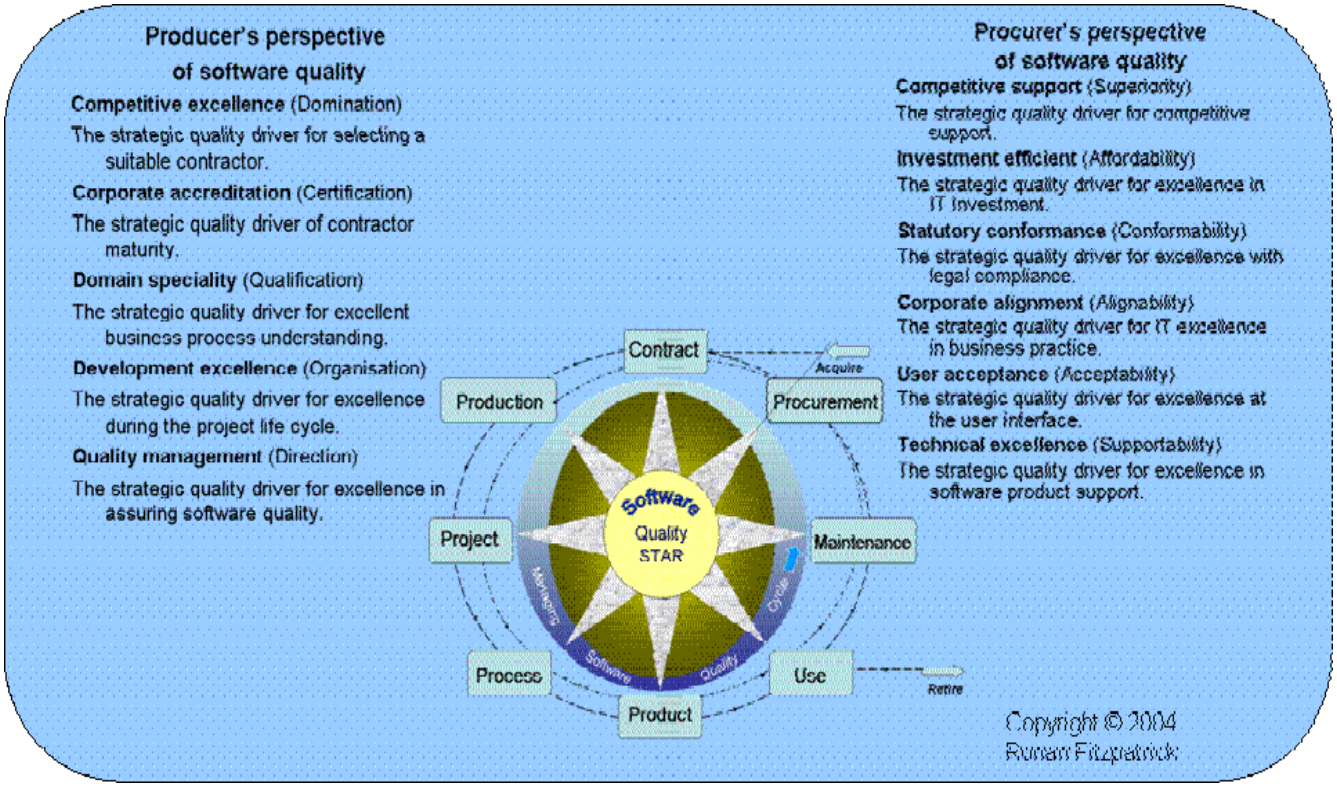
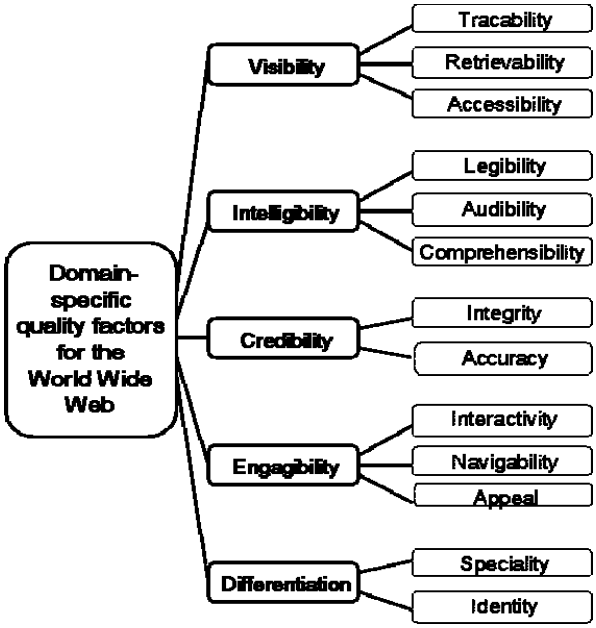


Fig.6. *Taxonomy of domain-specific quality factors for the World Wide Web (Fitzpatrick, 2000)*



The study of quality in the domain of the World Wide Web highlights new challenges as technology evolves – other domains will have different quality factors. For the WWW the challenges include methods and metrics for estimating, managing quality during the product life cycle and quality-of-use measurement. They will also include new emphasis on creating sites that support quality-of-ownership.

In their paper Software Quality Revisited address challenges relating to Web site quality. They address interpreting the Strategic Drivers in relation to quality Web sites and they also address the need for measurement methods and metrics in this domain.

6. Conclusion

The current paper summarizes the experience gained from an outsourcing project. The evaluation of the SQA and Project management programs has been made and some suggestions for their improvement has been proposed. Our future research will be directed to standardization of the proposed procedures thus facilitating their practical use. We will try to find a small and not very sophisticated project suitable for distant development so as to examine and approbate the feasibility and the usefulness of our new approach.

This paper also has set out a number of challenges which face those interested in software quality. These include:

- » A definition of quality which focuses on measuring excellence.
- » That interpreting the term usability as meaning anything that impacts the end user is a more natural interpretation of the term usability.
- » That the broader view of quality dictates that a life cycle that focuses only on software development is insufficient and that a full end- to-end software product life cycle is required as illustrated in the Software Quality Star mark II.
- » The expression Quality Assurance does not fully address the need for quality management throughout the product life cycle.
- » New challenges are presented by the need for quality of WWW solutions.

7. References

1. Applying software quality assurance to outsourced software development. http:// www.testlabs.com/white_papers.html
2. Boëhm B (1978) Characteristics of software quality. Vol (1) of TRW series on software technology, North-Holland, Amsterdam, Netherlands.
3. Crosby PB (1984) Quality without tears, McGraw-Hill books, NY, USA, pp:60.
4. Deming WE (2000) Out of the crises, MIT Press, Cambridge, Mass., USA, pp: 168-169.
5. Eskenazi A, Maneva N, R Radev (2000) Project management and quality assurance in a distant software development project. Proceedings 5th SQM Congress, Bonn.
6. Feigenbaum AV (1961) Total quality control: Engineering and Management, McGraw-Hill, NY, pp:13.
7. Fitzpatrick P Smith and O'Shea B (2004) Software quality revisited, Proceedings of the software measurement european forum (SMEF 2004, Rome), Istituto di Ricerca Internazionale S.r.l., Milan, Italy, pp: 307-315, ISBN 88-86674-33-3.
8. Fitzpatrick R (2000) Additional quality factors for the world wide web. Proceedings of the second world congress for software quality, Yokohama, Japan, Union of Japanese scientists and Engineers (JUSE), Tokyo, Japan.
9. Fitzpatrick R (2001) Strategic drivers of software quality: Beyond external and internal software quality", second asia-pacific conference on quality software, Proceed. APAQS 2001, Hong Kong; IEEE computer society press, California, USA.
10. Fitzpatrick R (2003) The Software Quality Star: A conceptual model for the software quality curriculum, workshop paper, Proceedings of closing the gaps: Software engineering and human-Computer Interaction, INTERACT 2003: Ninth IFIP TC 13 International Conference on Human-Computer Interaction, September 2003, Zurich, Switzerland.

11. Genichi Taguchi Elsayed A. Elsayed and Thomas Hsiang (1989) Quality engineering in productions systems, McGraw-Hill, NY, USA, pp: 2/3
12. ISO/IEC 12207 (1995) International Standard. Information technology-Software life cycle processes, International Organisation for Standardisation, Genève, Switzerland
13. ISO/IEC 9126-1 (2001) International Standard. Software engineering-Product quality-Part 1: Quality model, International Organisation for Standardisation, Genève, Switzerland.
14. Juran JM (1989) Juran on leadership for quality, Free press, NY, U.S.A, pp:16
15. Kaoru Ishikawa (1985) What is Total quality control? : The Japanese way. Prentice Hall, Englewood Cliffs, London, UK, pp: 44/5.
16. Maneva M and Maneva N (1994) Software product ranking. Proceedings 23th spring conference of the UBM, (1994) 238-244.
17. McCall J, Richards P and Walters G (1997) Factors in software quality. Vol I-III, Rome Aid Defence Centre, Italy
18. Oakland JS (1993) Total Quality Management, Butterworth-Heinemann, pp:4.
19. S(higeo) Shingo (1986) Zero Quality Control: Source inspection and the poka-yoka system, productivity press, Cambridge, Mass, USA, page vi.