STUDIA UNIV. "BABEŞ-BOLYAI", INFORMATICA, Volume I, Number 1, March 1996

SOFTWARE QUALITY MANAGEMENT: TO UNDERSTAND AND TO INTRODUCE (II)

E. BALLA

Abstract. Following a brief exposition of the history of software-quality definitions, today's most popular approach to the problem was presented in Part I: building quality management systems according to ISO 9000 Series. Some advantages and disadvantages of this trend were shown. In this part, using results of software quality oriented research and practical training done by the author in Hungary, some problems and their possible solutions are sketched to start introducing software quality management. The problem are similar in most Eastern European countries, therefore we presume that presented solutions could be applicable as well.

1. Software quality management

Software quality management is described not only in ISO 9000-3 [1], but in many other standards too: IEEE, EN and some more Military Standards.

Software quality managers seem therefore to have an easy job developing and maintaining quality management systems: theoretically they only have to read the many standards and apply them. But things are far from being that simple. Standards provide prescriptions for quality system elements and quality control elements – in general terms –, but they don't give the sequence of the activities to be done, in fact they let the quality manager guess not only how to do things, but also what exactly to do in order to develop a quality system that fits ISO requirements. If the work is done by experienced people, the prescriptions of standards are probably of real help, but unfortunately this isn't the typical situation.

In the following we present some problems in developing and introducing software quality management systems and try to sketch some solutions for solving them. We

Received by the editors: December 15, 1994.

1991 *Mathematics Subject Classification.* 68M15, 68M20, 68N99.

1991 *CR Categories and Descriptors.* D.2.9 [Software Engeneering]: Management Software – quality assurance; K.6.2 [Management of Computing and Information Systems]: Software Management.

[1] Guidelines for the application of ISO 9001 to the development, supply and maintenance of software

proceed by taking into account the Hungarian situation, which can be considered typical for most Eastern-European countries.

1.1. **Before ISO 9000.** For a better understanding of the present situation, we have to consider the *software-quality oriented research and work done before ISO 9000 appeared.*

We can state that despite the former isolation of Eastern-European countries, the quality-research in these countries has followed the steps of Western development. It's worth to note that the first attempts for identifying software quality attributes and their relationships were made in 1977 in Hungary – that means 7-9 years of delay in comparison with the USA. Western-European situation has also to be taken into account: SIEMENS, for instance proceeded only in 1975-76 in identifying these elements, that places Hungary at two years of delay. In 1977 in Hungary Basic Quality Requirements for software have been stated, and by 1985 some quality-measuring programs also have been developed. These attempts fit in the world-trend of measuring software quality through some well-identified attributes and in application of the product-based definition of software quality.

1.2. **What's happening now – and what can be done.** Like other companies in business life, Eastern European software companies join the trend of developing quality management systems according to ISO 9000.

1.2.1. *Quality organizations in Eastern Europe.* In the Eastern European countries national quality organizations have been organized, which have insured the translation of ISO 9000 standards, are organizing conferences, meetings on quality-related themes [2]. In most cases there are foreign registering organizations working in these countries [3]. National quality organizations and research institutes also get more and more envolved in enhancing application of ISO 9000 under the spacial conditions given in each country. The *Processus* project, for instance, founded in Slovenia, supported by 11 Slovenian software companies, has the scope of developing a tool for auditing and goal definition. (The project and the tool are presented in [Gyorkos94]. *Tipically no software company has been registered yet.*

[2] In Hungary, for instance, in 1993 there have been about 8 quality conferences – the largest being the annual Hungarian Quality Week – each of them involving the participation of about 80-100 companies

[3] In Hungary: Bureau Veritas, TÜV Rheiland, TÜV Bayern, TÜV Hannover, SGS (foreign), and a Hungarian registering organization (Mertcontrol)

1.2.2. *Teaching quality management.* As we briefly presented above, ISO 9000 operates with many quality-related terms, which have to be *well understood* and *correctly applied*, as in other fields of working and academic life. According to [Geiger93] in the field of quality management this "necessity is of even higher importance because of the lack of basic knowledge from education".

In many Eastern European countries quality management is taught only on off-school courses, which, for the moment, are too expensive to afford for individuals and for small companies (a 2-day course costs about 650$). Consultancy alone for developing a software quality management system costs about 30.000$, therefore such systems are developed mainly by "self made" quality managers, who, in best case, have joined some quality-oriented conferences and read the available theoretical material. For these people *standard's definitions are often not relevant.*

At some universities (Technical University Budapest – Hungary –, Technical University Maribor - Slovenia - for instance) basic project management and software development methods are being taught. *Restructuring the quality-related educational system* would be a first step to change the present situation. Postgraduate courses have to be replaced by usual educational elements, first in universities, than in high schools too. Some models are available in western countries[4].

2. Developing software quality management systems

The first step in developing a (software) quality management system is **identifying the present situation**, in order to sketch what is to be done.

We try to describe the present situation and discuss some elements that can prepare introduction of software quality management according to 4 problem-groups, which, in our opinion, are basic when a software quality management system is to be developed.

These groups are:

a. *Applying a structured methodology for system design and development*
b. *Applying a project management methodology*
c. *Organizing other (non-project-related) activities at the company*
   i. *services*

external (e.g.,courses, consultancy)

[4] Zulema Lopes Pereira [Pereira93] presenting the Portugaal quality-situation mentioned that Portugal's national quality policy hass been stated in 19911, and in 1993, every university student had weekly at least 3 hours of quality management courses

internal (e.g.,network maintenance)

ii. *information system*

iii. *human relationship*

d. *Writing down aspects regarding the elements a.–c. in a clearly stated and easy-to-understand manner*

2.1. **Project-related activities.** In the following we present some problems connected with the first two elements (a.-b.) of the above described group.

The *structured system developement and project management methodologies* started to develop in the eighties. These methods and methodologies decompose the system development activities – structure them – into stages, phases, steps, define the sequence of the activities. They prescribe precise inputs and outputs for each step. They recommend a top-down approach and a step-by step refinement of the system. They make a strict difference between logical and physical planning. The techniques used are similar. (Entity – relationship modeling, function hierarchy modeling, entity – life – history modeling, data-flow modeling etc.). Nowadays they cannot be used without complex computer-aided tools, which help to transform the logical model into physical model and even into program code[5].

2.1.1. *Structured system development methodologies.* SSADM (Structured Systems Analysis and Design Method) has been developed in the UK by LBMS (Learmonth and Burchett Management Systems), by request of CCTA Central Computer and Telecommunications Agency for a self-testing, teachable methodology, which would use tested system development methodology in the UK[6].

Oracle Case Method* was defined in the early eighties by Richard Barker, the first book about the method was printed in 1989. It has been derived from many years' collective experience delivering systems to clients and building software products.

The two mentioned methodologies differ in the life-cycle-steps covered, in their most recommended techniques, in their notations.

[5]More and more structured system development and project management methodologies become available in Eastern European countries. Presentations in sections 2.1.1 and 2.1.2 does not suggest any hierarchy among these, it just deals with metodologies considered by the author being the most used.

[6]SSADM has been translated into Hungarian, and it is the metodology most probably becoming recommended in Hungary

Oracle CASE* Method	SSADM
Strategy	*Feasibility study*
Analysis	Problem definition
Design*	Project identification
Build	*System analysis*
User Documentation	Analysis of systems operations and curent problems
Transition	Specification of requirements
Production	Selection of technical options
	System design
	Data design
	Process design
	Physical design*

TABLE 1. Project life cycle in two methodologies

Table 1 makes a comparison between the project life-cycle elements covered by the SSADM and Oracle Case* Method.

To be noted that neither SSADM nor Oracle Case* Method cover the entire project life-cycle. A possible model for covering all life-cycle should be worked out by each company, according to the work's characteristics.

2.1.2. *Project management methodologies.* *PRINCE* (Project Run IN Controlled Environment) is a project management methodology – in fact *the* Project management Standard of the UK government IT departments. The methodology consists of 3 major components that are applied to a project: ***organization, planning*** and ***control.*** Using it will lead to a common understanding of responsabilities, of activities, of terms, an efficient and familiar documentation and will reduce dependance on the individiual.

TickIT was prepared by the British Computer Society under contract to the UK Department of Trade and Industry. It isn't exactly a system development nor a project management methodology. It's a guide to software quality management system construction and certification, using ISO 9001. Purchasers' guide, suppliers' guide and auditors' guide provide guidance for quality system elements and quality control elements which can be used not only in developing a software quality management system, but in defining system development and project management steps too.

2.1.3. *Software tools for structured methodologies.* The real help for using such structured methodologies is given by their software tools.

SSADM Engineer is the software tool-set developed by LBMS for supporting SSADM methodology. The current version (SSADM Engineer 5.0, September 1993) runs on Windows 3.1. It operates with software prodeced by Gupta Technologies Inc., using a database for maintaining textual and graphic information about the system to be developed. It supports almost all life cycle functions of SSADM, providing help in transforming logical model into physical model.

Using Oracle Case* Method is supported by the *Oracle CASE* tools, which facilitate the interaction with data maintained in the central database, implemented with the Oracle SQL data base management system. *Case* Dictionary* and *Case* Designer* provide defining and repositoring the data used in logical planning (textual and graphic), while application-generators, such as *Case* Generator for SQL* Forms/SQL* Menu, Report Writer* provide transforming the logical design-related data into running program components.

Project management methodologies are supported by varoius software tools, such as: *Microsoft Project, team Windows, Artemis Prestige.*

Artemis Prestige was developed by Lucas Management Systems. It runs under Windows, co-operating with many types of hardware, using Oracle, Gupta SQL Base database management systems. It provides help for project planning (timing, resources, cost-analysis).

2.1.4. *Using structured system development and project management methodologies.* The first two elements in the problem group presented at the beginning of the chapter (a.,b) seem to be easy to realize. Apparently they would "only" require a management decision regarding the use of a structured system development and a project management methodology. At software-companies the use of computer-aided tools is not a real difficulty.

However, the situation is much more complicated. The majority of analysis and programmers have their "good old" methods to develop a system, and in many cases the customers also have some requirements – differing one from another – which have to be satisfied. So, instead of applying one structured methodology for the whole project, teams rather apply *some elements of certain methodologies*, combining them with others

to satisfy customers' needs. Project managers in most cases are aware of the advanteges of using a single structured methodology, and so is the company management. The main reason that impedes consequent use of a methodology is time – that means money. Companies have to live, and – for instance – SSADM methodology with all its steps, tasks and documents is considered to prolong system-development by such an amount of time, that cannot be afforded by a small company.

It is very important to identify *product life-cycle and system development steps.* While strategy, analysis and development steps are followed in the majority of cases, *project preparing steps* – offer, tender problem definition, first-cut project plan – and *product maintenance steps* – installation, maintenance, analysis of the errors, maintaining user observations, etc. – *are often neglected.* They are not considered to be project elements, therefore time for doing them isn't allocated. Project members have to do these activities "outside" the project, many times parallel to an other project. The rush has negative effects on the success of the project – *the contract is not well-defined, the reasons of success or failure of offers, tenders are not analyzed, user-reactions, observations are not followed and analyzed*, etc. *Marketing plans, cost analysis are seldom done*, and seldom by experts. *Test-plans, configuration* and *change management plans are not being used*, although every project manager feels their lack. Usually *there are few checkpoints in the projects* when the team could discuss its problems with the management.

Users are seldom cooped into reviews – in fact they are only cooped when they insist on it, projects evaluation reviews are seldom organized.

Applying a structured system development and a project management methodology, in my opinion *cannot be achieved at once.* If someone tries to convince system analysts and programmers about the fact that their working methods are bad, they will demonstrate in short time that prescriptions of the quality management system lead to worse results. Nevertheless, there are some elements that we can start with.

Being aware of the lengthening of software product life cycle would be one of these. Project initiation steps and follow-up have been done, the results of tenders and users' observations have to be analyzed and the experience used in forthcoming projects.

Using a structured system development methodology and a project management methodology can also *be introduced step-by-step. SSADM, Oracle Case Method* with their computer-aided tools offer, first of all, an organized working frame. Getting familiar

with their concepts – even without using them – will lead to a change in the mentality of system-developers. They shall notice, for instance, that both models make a strict difference between logical and physical design and prescribe precise inputs and outputs for each step.

Organizing project-related data is possible at any moment. *Recommended directory structure* of all data related to a project can be described, *archiving methods* can be worked out and documented. Employees are to be encouraged to keep, for instance, their project-related incoming mail (electronic and ordinary) separately from the project oriented outgoing mail and from other subject mail.

There are some elements in *project' quality management* that can be applied from the beginning. Project managers, programmers will notice the help given by *configuration management plans, change management plans, test plans* and *test journals, journals of project related activities* or *list of documents used and prepared.* They will find – at first informal – *quality reviews* useful. *Applying corrective actions during a stage, not after finishing it is a basic goal of quality management.*

Project documentation is an important part of project-related activities. The problem is here that documentation in a company is seldom uniform, every team develops its documents according to the needs of the customer. The quality management system has to prescribe the outline and the content of each document type, so, the many types of documents have to be gathered in a uniform system, that fits the needs of every user and every project member.

2.2. **Non-project-related activities.** The problem with the activities sketched in point c. of the problem-group presented at the beginning of this chapter is that they seem to work. Announced courses are held, users' problems are solved sooner or later, system backups are being performed more or less regularly. But, because of poor organization these activities imply much more effort than normal. When a project manager, a system engineer or a secretary goes on holiday, it takes a real effort for someone else to handle his/her job. It's a big problem, when in an office a computer has to be replaced, because no one knows for sure the configuration, the utility-versions that have to be applied.

Such difficulties can be surmounted by appling a well-organized *information system.*

The information system – which is the totality of electronically and manually preserved information – can exist before a quality management system is being developed. In most cases this system is developed together with, or in the best case, it is documented as a part of the quality management system.

Building the information system can start with unifying the documents and documentation used by the company not only in project-related activities, but in any other activity done. Templates should be described not only for project documentation, but for official letters, fax-headers as well.

Employees have to be noticed about every new feature of the "nascent" information system and have to be encouraged to use them and report any connected objections.

Human relationships have an important effect on the company's work. If the working conditions are good, the efficiency of work is higher. Employees have to carry out tasks they are professionally prepared for. Informatics is an even-changing science, so there is no shame in not knowing something. But people are sometimes put to do things are not prepared for, therefore they work much harder, are stressed-they want to prove they fit management's trust–, therefore their working capacity decreases.

Services, other, non-projrct-related activities can be described only in co-operation with people who effectively do the related work. If tehere are some rules, the quality manager should describe them, if there are no rules, he should work them out together with the system engineer, training manager, secretary, etc.

Communication among team members, management relationship with employees has to be organized without being unnecessarily formalized.

Human relationships cannot be changed by the quality management system. However, there is a hope that well-organized activities will diminish stress and extra work – which lead to an improvement of human realtionships at all levels.

2.3. **Writing the quality management system documentation.** At last, but not least, writing the quality management system documentation is an important part of the system. We've mentioned it in point d. of the problem-group presented at the beginning of this chapter because we think it's better describing an existing system than trying to build up the system according to the previously written documentation. Assessment for

registration begins with consulting the documentation, so we could be tempted to write a "perfect" documentation set, according to ISO 9000 prescriptions but without taking into account the existing situation. Such a quality management system cannot be viable.

3. Conclusions

Due to the world-wide acceptance of ISO 9000 Series, we speak more about developing, introducing and maintaining software quality management systems than about the quality of the software product, as defined earlier.

Basic motivation for developing such systems is the wish of the software companies to get registred according to ISO 9000.

With all desire to get registred in the shortest time, I think software quality management systems cannot be applied at once after being developed. Applying every prescription would entirely change a company's activity, causing financial and more crisis.

The best thing that can be done is to involve all employees in the development of the quality system. The management should present the concept as necessary for the company's welfare, the use of which is well motivated. It is recommendable that the motivation should't be ISO 9000 registration, but facilitating employees work.

However, applying the elements described above seems to be possible. These elements are accepted easier probably because of the immediate help they provide both in system development and project management.

References

[Beiczer78] Beiczer, Ö, Szentes, J. Felhasználói programok forráslista alapján történő mininősítése, Softtech sorozat, D26, SzKI, 1978.

[Boehm78] Beohm, B.W., Characteristics of software quality, TRW Series of Software Technology, Vol.1, North Holland, 1978.

[Develin93] Develin & Partners Management Consultants, Freeing the Victims – Achieving Cultural Change Through Total Quality, 1993.

[Feigen93] Feigenbaum, A., EOQ World Conference, Helsinki, in: MMT Tarsasagi Tajekoztato, II. evf. 12. sz. 1993. dec.

[Garvin84] Garvin, D.A., What does "product quality" really mean? Sloan Management review, Fall 1984.

[Geiger93] Geiger, W., Talk Show with ISO 8402? Contemplative thoughts about the new quality vocabulary, EOQ Quality 1/1993, pg. 13-17.

[Genuch91] Genuchten, M. van, Towards a Software Factory. Ph.D. thesis, Eindhoven University of Technology, 1991, ISBN:90 900 4119 2

[Gyorkos94] Györkös, J., A support for auditing the software process, in: Austrian-Hungarian Seminar on Software Engineering Klagenfurt, 7-8 April 1994, pg.7.1.-7.4.

[Havass93] Havass, M., A szoftvervizsgálat múltja és jövõje, II. Minőségi Hét, 1993, November 8-10., A konferencia előadásai, II. kötet, pg. 85-101.

[Interl93] The ISO 9000 Guide, Interleaf, 1993.

[McCall78] McCall, J.A., The utility of software metrics in large scale software systems development, IEEE Second software life cycle management workshop, August 1978.

[Oracase] Barker, R, Case* Method, tasks and Deliverables, Addison-Wesley Publishing Company, Revised Edition, 1990.

[Pereira93] Pereira, Z.L., EOQ World Conference, Helsinki, in: MMT Társasági Tájékoztató, II. évf. 12. sz. 1993. dec.

[PRINCE91] Project Run In Controlled Environment, LBMS Delegate Notes, 1991.

[SSADM88] Downs, E., Clare P., Coe I., Structured Systems Analysis and Design Method – Application and Context Prentice Hall International Ltd. 1988.

[Stephens93] Stephens, K., Quality Systems and Certifications – Some Observations and Thoughts, EOQ Quality, 1/1993, pg. 5-12.

[Szentes82] SOMIKA – An automated System for Measuring Software Quality, Proc. of Premier Colloque de Génie Logiciel, AFCET, 261-276 (1982).

[Szentes83] Szentes, J., Qualigraph – A Software Tool for Quality Metrics and Graphic Documentation, Proc. of ESA/ESTEC Software Engineering Seminar, Noordwijk, The Nethderlands, 73-81, 1983.

[Szentes85] Szentes, J., A szoftverminőség és mérése, Számitástechnika-alkalmazási Vállalat, Budapest, 1985.

[TickIT90] Guide to Software Quality Manegement System Construction and Certification using ISO 9001/EN 29001 Part 1 (1978). Issue 1.1 (30 September 1990) British Computer Society UK Department of Trade and Industry.

[ISO9000] Quality management and quality assurance standards – Guide for selection and use, 1987.

[ISO9001] Quality system – Model for quality assurance in design/development, production, installation and servicing, 1987.

[ISO9004] Quality management and quality system elements – Guidelines, 1987.

[ISO9000-3] Quality management and quality assurance standards – Guidelines for the application of ISO 9001 to the development, supply and maintenance of software,1991.

Technical University Budapest, Department of Mathematics and Computer Science, Group for Computer Science and Information Theory

E-mail address: balla@inf.bme.hu