Product Quality & Software Quality

Lélis Tetsuo Murakami, Eng.
Delphi Diesel do Brasil Ltda. - Consultant

ABSTRACT

The industries have been working hard, for decades, to improve quality in their products, using process that assure a quality level which meets customers needs.

An important and strategic component, the IT area, collaborates with the companies to meet the goals satisfactorily, which is not considered with the necessary attention. The IT area of an organization should be managed with the same importance dedicated to product manufacturing process.

The object of this paper is to convince the industries that the desired quality in products depends on more and more of IT resources and because of it, is strategically important an adequate monitoring of IT quality, based in corporative governance.

INTRODUCTION

The quality in product and the quality in software are assured by similar process but differ in many aspects. The product is easier to be specified than software. The action on a product is clear and well defined but, for the software however, the solution could produce different results to the same requisition. That is because software depends on the development platform, the language used, the interface projected, the programmer experience, etc. As a consequence, we get software with different levels of quality.

Product becomes older and its components could be changed, making possible an extra life. Software stays the same along the time, but become older in terms of obsolescence whose recovery is onerous, in general, and requires a new re-writed version, in most of cases.

To well understand software quality, it is convenient to make an investigation about the history of quality in manufactured products. At the end of 60 decade in Japan, was created, may be the most efficient one, a process to specify the set of requirements desired to a product or service, according to the customer point of view. This method named QFD – Quality Function Deployment is part of the TQM – Total Quality Management tool. During the decade of 1970, this technique was improved and became a very strong and reliable tool, spreading across the world, more significantly in the USA in the following decade. This method aims essentially to collect and understand the customer needs or expectations about a specific product, transforming them in a list or requirements used along the product manufacturing process. As a result of this strategy, a reliable product with desired quality is obtained providing the customer satisfaction. QFD goes more deeply trying to find out more requirements not related by customers through interviews, but inferred during the process, in order to provide some characteristics to the product that will cause a positive surprise to customers, a very useful strategy to improve the customers satisfaction. The same process could be used to establish quality in services. This method could be applied to software development, but there are others alternatives that will be discussed in details in following chapters.

THE PRACTICE OF QFD

QFD history was originated in the late 1960 and early 1970 in Japan. At that time, the Japanese industries were performing the "copy" manufacturing that is, use existing products, to create another one. In this status quo, QFD concept was created to new product development. Akao & Manzur in [1] describe in details the history of QFD evolution referring to the past, present and future of this method. Akao after created this method, observed these two points:

- The importance of quality in the product manufacturing is a fact. However there was no available bibliography about the theme and how it could be provided;

- The manufacturers had already used worksheets in the quality control process but after the product manufacture.

- and concluded with the question:
Why could not we note the critical points of quality control process prior to production start up?

The term QFD – Quality Function Deployment was originated from the Japanese term 品質機能展開 (ひんしきのうてんかい) or (HIN-SHITSU KINOU TENKAI). The translation of this term was modified by the author along the time. In 1972, Akao translate it as Quality Function Evolution and in 1983, Masaaki Imai suggest other translation arguing that TENKAI would be better translated as Deployment, and Akao agree with him.

QFD method was spread significantly through the USA and other countries beginning at the 80 decade. A survey conducted by Tamagawa University and University of Michigan on recent QFD applications shows that 68.5% of American companies use QFD in their activities and only 31.5% of Japanese companies use it. Yoshizawa in 1997 points to a relevant aspect of the QFD technique:

- QFD had modified the quality control in the product manufacture, changing the direction of the quality control to the development and project. The quality assurance QA, which was before oriented to the process, now is oriented to the project, creating a product development system.
- QFD constitute a communication way between designers, that is, between engineers from Marketing and Production areas.

There are many issues in the literature about QFD and its application in product manufacturing. Lai-Kow Chan and Ming-Lu Wu [2] made an important work compiling many articles about the subject, revising 650 papers about QFD from several sources. In this paper the authors describe the evolution of QFD in Japan and present a list of QFD organizations, software and on line resources. In the sequence, the authors make a functional analysis of QFD, comment industry applications and show the evolution of the method.

The bibliography of QFD applied to software development is poorly related. There are differences between to produce products and to produce software as well as there are differences to specify products and specify software. More over, a customer requirement of a product is clear however, it could be subjective in case of software. A software could attend to a process need and this later attend to user. In this case we can say that the user is attended indirectly and so, the QFD method could not be applicable easily. Barnett & Raja in [3] justify the few bibliography of QFD applied to software development arguing that the companies after using the method obtains significant market advantage and so, they maintain the QFD experience proprietary, hiding the knowledge to own use.

Other authors such as Mercedes Ruiz, Isabel Ramos and Miguel Toro in [4] assume a different point of view referred to software development process suggesting a strategy of quality assurance, based on modern practices of system modeling. The method described in this paper uses a traditional software development method joined with extensive use of a dynamic simulation model of the development process in order to obtain a quantitative and qualitative base to improve the software development and subsidize the decision make process reaching high values of capabilities of software development according to CMM – Capability Maturity Model.

QUALITY IN SOFTWARE

A software as a product could be considered in the same way as a manufactured product. Its quality could be perceived as how it performs his functionalities with efficiency under the customer point of view. Although this, the list of functionalities could be large and so, one difficulty is to guarantee that all of them and its combinations will work. The traditional software development process suggests the use of several strategy of test to verify if all functionalities are working. However, the application of this procedure becomes very hard when the number of functionalities or its combinations assume numerous values. Therefore, the guarantee of software quality is inversely proportional to the number of functionalities or its combinations.

If a software should sum 2 numbers, probably will do it well presenting a quality of 100%. However the Windows Vista system, will have several operational difficulties as predictable. Certainly this software was made by groups of many software engineers and programmers and the probability of problems occurrence is high and depends on the software gigantism. Although the problems exist, it is not correct to imply that the quality was refused. In fact, what is reasonable is to adopt a strategy considering some parameters like cost, time, quality levels, test levels, reliability, etc. It is important to understand that quality improvement requires financial resources available. Referring to software development, there is another point to consider, that is, if some software has a wide spread in the organization and has poor quality, it will cause undesirable damage as a presumable consequence. The good strategy is like that: First, do make the customer satisfied. Second, we should adopt a quality level according to the resources available. Finally, provide some efficient procedures like good after service and customer assistance to repair the eventual problems caused by the insufficient quality level avoiding to loose the customer. Wide projects require rigid patterns, good communication through clear and concise language and a strong administration. Even though these points were provided, it is not sure that the software will work with no problem. It is impossible to assure the total assurance that the software will work as designed. One
strategy concerning to this problem is to assume certain level of quality and reliability through various ways to support customers in case of faults, providing quick and efficient procedure to help them. Many customers would rather products with not so high quality but supported with a very good after service providing immediate solutions to problems than others excellent products with bad services, taking days long to solve problems. A common practice adopted by software industries is to assume a certain level of quality and develop a version for delivery. After that, according to user reclamations, they fix the software and deliver corrections packages through Internet with no charge. Frequently, users are customers and they test the software free of charge, document the bugs and send it to software developer. Depending on the spread of the software around the world, million of users are working to find out the bugs.

If there is no possibility to inspect all range of software functionalities there is a strategy to guarantee that part of it will work correctly. This idea is quite old and means "divide and conquer" that is, let us divide the problem in sub problems and try to inspect them. A software component is part of all, which process a specific function and has a generic utility. There are many examples of software components that have high level of reuse.

There is no doubt that there are enough differences between products and software manufacturing making the use of QFD impracticable in completeness. However many concepts of this technique are useful, mainly to identify the requirements. The matrix HOQ – House of Quality could be used partially to identify and prioritize the software requirements under the user point of view. Although the wide spread and success of QFD applied in the industry, this not means that this methodology could be applied in software development with the same success. More than this, it is not true that the software development will be damaged if QFD is not applied. Software engineering and OO methods always had aimed the excellence in software design. These techniques working together had provided quality in software development through the large use of methods, patterns and management process. The strategy of investing in software development process to obtain quality seems to be adequate and the most successful.

**PATTERNS AND REUSABILITY**

Software quality is a characteristic that could appear differently and depends on several factors. The quality level established would be a strategic parameter. However, some procedures should be adopted in order to maximize the quality level.

The use of patterns in the software development process has several positive aspects. Through patterns, the development process got high performance because no time is lost to think and decide which style will be used or what language will be selected or how the code should be developed. The more patterns are clear the more high performance is obtained. When we talk about patterns it is convenient to select tools and communication language to promote the link between all actors of the development process. Well done documents of system, follow-up, failure statistical analysis, support, observed failures, etc. are necessary to manage the organization process. Coding patterns also drive to improve programming activities and certainly promote the economy of many hours of maintenance.

The most important advantage of using patterns is the transparency of the development process, making this activity independent of the programmer. It is strongly recommended to adopt a hard administration to implement a development procedure based in patterns. If there is no adequate control, the results obtained could be inefficient. The IT area has becoming one of the most important and strategic department of ever organization which intends to be competitive in the market. Actually it is not imaginable a company without a strong IT department manipulating operational data and providing information to the company’s executives, and feeding the DSS – Decision Support System of the organization. To improve the risk analysis process and the decision making, a structured process is necessary to manage and control the IT activities of the company in order to guarantee the investment payback and improve the organization process. This new IT wave is called IT Governance. There is no doubt that the IT technology is needed with all of its paraphernalia such as PC, networks, hubs, servers, internet, firewall, etc. The point of discussion is how each company manages his technical tools. That is why model like ITIL – Information Technology Infrastructure Library has becoming widely popular in the IT world. However, few companies have adopted it, but the concerning exists and the diversity of methods is a fact. ITIL is a collection of best practices and process of IT services and was created in the 80 decade by CCTA, an agency of Britain govern. It covers many areas but has the focus in quality of IT services management and if well applied improve the efficiency in the IT management. Another model called CobiT is a guide to the IT management proponed by ISACF – Information Systems Audit and Control Foundation, www.isaca.org. The CobiT practices are recommended by IT specialist because it permits the optimization of investment and provide a way to measure the results. CobiT is platform independent and is oriented to business providing detailed information for business oriented objectives. We must point to another methodology for IT management called Rational Unified Process – RUP. It is a software engineering process and has a main characteristic of delegating tasks and responsibilities between the development group or development company. The main purpose of this method is to guarantee the production of software in time with planned cost and with the quality
required by the customer. The RUP development life cycle is divided in four phases named: Inception, Elaboration, Construction and Transition.

CMMI – Capability Maturity Model Integration is a reference table used to evaluate the software development maturity. The SEI – Software Engineering Institute from Carnegie Mellon, USA, since 1986 has developing and improving this evaluation table, which first version was called simply CMM – Capability Maturity Model for Software. The objective of this model is to provide for the organizations a method of measuring maturity in the software development process as well as to establish continuous improvement programs. We can define maturity as the capacity of replicate the success now obtained in future projects. All these methodologies focus strongly the IT activities management and as a result we obtained the assurance of quality in software. As much the companies structure themselves and adopt methods and patterns the probability of component reuse will grow up. Development strategies that focus the software development continuous improvement will reach quickly the high level of CMM.

SOFTWARE QUALITY IMPLEMENTATION STRATEGY

The first point that arises when software quality is been considering is the strategy to be adopted. This later could be unique in the organization or not but one fact is real and doubtless: assuming maximum quality and reliability for projects are impracticable instead, good quality and good reliability is better. To illustrate this idea we can think in an aircraft which risk of accident we want to establish as zero. In this case this plane should never fly and obviously, this is undesirable. The reliability value usually stated in commercial aircraft is less than $10^{(-9)}$ failures per hour. What can be considered are levels of failures which should be manage like in railway and aviation where failures are projected to assume small values. How to guarantee reliance eliminating failures? There are many alternatives that should be analyzed carefully before adopting one of them as the more recommended. In terms of customer products maybe the best strategy is to satisfy them. In this aspect, QFD becomes an excellent tool to analyze and prioritize the customer requirements and if possible, surprise them with good features or quality as recommend the QFD practice. One more point is important and should be considered: assuring a certain quality level requires time and resources and referring to software, we must consider the available time to develop it. In many cases, the available time for software development commands the quality level and in the same way, available resources would establish the quality. In any case, despite all these difficulties it seems to be important to maintain the user requirements with no changes. The strategy to avoid these difficulties could be managed through taking good planning, methods, improvement of performance, components reuse and all kind of efforts to maximize productivity and minimize the consequences assuring the quality level projected. Actually, companies should provide good productivity with adequate quality level as a way to become competitive. One strategy to implement quality in software is to invest in efficient IT activities management such as CMMI, PMI, COBIT, RUP, etc. as cited before which provide the necessary organization to execute software development activities with patterns and methods. The use of UML – Unified Modeling Language is efficient and promotes good and reliable communication between the actors involved in the development process which makes the quality level up. As pointed out before, quality requires resources and what should be done if a company does not have enough resources? Working with no quality strategy is out of question. However, a good strategy could be made as the following:

a) Use existing IT management practices defining actors, roles, disciplines, etc.

b) Describe advantages and disadvantages of the application of each roles and cost associated.

Doing so, it is possible to analyze and search what roles the company can perform, according to its available human and financial resources, assuming the risk of non executed activity and controlling the consequences. This is preferable than quality ignoring strategy and will reduce the non desirable situation. The companies will win the market competition as successful as they invest in generic software components which are improved every time when reused, making the continuous improvement and transforming them in doubtless high quality software peaces. The more components are created, the more improved will be the application development. In a short future, people will assembly software instead of coding it. Many templates will help the developer to quickly create reliable applications using powerful wizards in a very short time. Quality in this scenario is an inherent characteristic and nobody is worried about. One more question arises when we talk about software quality. Frequently hardware, software and peopleware are not equalized in terms of quality. I introduce here the concept of HSP normalization, that is, the idea of manage the compatibility in this three axes. It is a simple question but very often, developers forget this issue and make applications with unbalanced HSP. To normalize it means to increment conveniently the effort along these axes. For instance, it is not necessary to use parallel processing hardware if the software does not use parallel skills in the software coding. What to say about a complex algorithm that only the author could understand and use? What the need to research and implement a very complex algorithm to get the optimal solution if the entry data have percent error of two digits? In this aspect, the quality is not in the software but in the good sense of developers who are supposed to be regardful to these considerations.
CASE STUDY

The issue quality of software development has been focused with considerable priority in an industry of automotive field in Brazil, belonged to a large international group. In 2006 this company had 800 employees, annual billing about US$ 76 millions. The IT infra structure was compounded of 20 servers monitoring Databases, Intranet, Mail, ERP, Applications, Metaframe and Product Engineering. The network connects 280 PCs and 30 laptops. The upper bound processor was Xeon 3.06 for servers and Pentium IV 2.8 for PCs, both using Operational System Windows 2000. The IT team has 15 professionals hired in outsourcing regimen where 8 of them are directly involved with software development. As part of an international group, the company administration should comply the group regulations and patterns and follow the recommendations resulting from American laws. The assurance of quality to this company was implemented considering the existing best practices of IT administration, customizing them to the company available human and financial resources including the compliance of internal regulations and Sarbanes-Oxley law (2002). This latter was created in USA to increase the corporate governance, given to directors more responsibility over financial controls and report delivery procedure.

The necessity of this project was cause by the objective of the company to regulate the development process which main guidelines are:

- To comply internal regulations of software development;
- To improve the performance of software development activity;
- To make the process of development transparent to everyone;
- To attend the project time established;
- To implement software quality.

The IT team of the company has used until now several methods to elaborate the software projects to supply the demand. It remains however, a difficulty related to the software development which did no have the same systematic procedure causing inefficiency in maintenance process and project control. This variation of procedures generates increases of rework and costs of maintenance and development. As the tests criteria and development parameters were established in different ways the evaluation of quality pattern and the attendance to the user requirements drive to mistaken judgments, mainly because the difficulty to implement a reasonable metric system. Although the procedures of development adopted by company were consistent there was a necessity to reorganize and take similar development procedures, based in current IT practices such as CMM – Capability Maturity Model and UML – Unified Modeling Language.

To elaborate the methodology, all the company internal regulations and procedures such as Sarbanes-Oxley, IPS-Information Policy System, Internal Control, etc. were analyzed focusing IT topics. In terms of practices, RUP – Rational Unified Process, PMBoK from PMI, CMMI, COBIT and ITIL were also considered.

The basic guideline for this project had consisted in the attendance to internal regulations and the compliance to the Sarbanes-Oxley law. The methodology developed was customized to the company real conditions referring to available human and financial resources. Considering these assumptions, the objective was to structure a methodology which makes possible the best IT governance. The strategy was to extract from the practices mentioned above the relevant points that provide to reach the aims.

The basic guideline for this project had consisted in the attendance to internal regulations and the compliance to the Sarbanes-Oxley law. The methodology developed was customized to the company real conditions referring to available human and financial resources. Considering these assumptions, the objective was to structure a methodology which makes possible the best IT governance. The strategy was to extract from the practices mentioned above the relevant points that provide to reach the aims.

Based in RUP methodology, we used the same concept of software project life cycle, dividing it in four phases called Inception, Elaboration, Construction and Transition, involving several disciplines. Each phase is compounded by various disciplines and each of this later is a set of activities. For each activity was defined the actors involved, their correspondent roles, the responsible actor and the artifacts to be constructed.

The disciplines considered were:

- Services request;
- Requirement Management;
- Project Management;
- Configuration Management;
- Development;
- Tests;
- Quality.

In order to afford more flexibility to roles attributions and activities executions for IT members, each of the activities was listed and described, mentioning its meaning, the actor who was designated to do it, costs involved and the results expected. Using all these information, the company can evaluate each activity and the correspondent cost to decide if this activity with this cost which will provide these results is interesting at that moment.

The quality requirement was intensively considered in the activity of system specification, using the concepts of...
QFD to identify the user (or process) requirements and transforming them in project specifications. The methodology established had used intensively regulations and patterns of software development process, improving the software quality. The reusability was also focused on this methodology, developing generic components when possible and creating the continuous quality improvement process.

The quality implementation in IT area, as well done in product manufacturing, consume significant amount of financial resources, in most cases not available, and this can cause some damage to the synchronization with the manufacturing. As a consequence, the IT quality will be decreased and because of it, many problems will appear in several areas of the company, requiring hard and difficult arrangements to maintain the current quality level.

It is mandatory to understand that the organization as a hole is totally dependent of IT resources and these should be considered as an important co-worker for quality goals reaching. More than this, the good ability of using IT resources provides a gain in performance, in terms of quality, shifting to an upper level, which drives to obtain a better level of quality.

Although we designed the structure of IT quality, its implementation is in stand by because of global directives established to the group. The company is involved in a strong movement of globalization, changing significantly the course of investments causing a momentary difficult. It is a good strategy to establish global directives adopting the same posture for all branches.

CONCLUSIONS

The concept of quality in software is not supposed to be considered as it is in product manufacturing because there are differences between producing a software or a consumer object. However, we should attribute the same importance in terms of efforts and resources because Product Manufacturing and IT work together and this composition will result in considerable increment in the performance of quality level.

It is not recent the fact of many organizations consider the IT area as an accessory that consume big amount of resources. The managers always have the difficult to understand the problems of this area: Why a development always consumes more resources than the budget allocated? Why the projects always cross the deadline? Why so many bugs? Why systems are constantly reviewed? These concerns normally are explained badly (or misunderstood) or seems like not convincing, and because of it, there is a tendency of managers to be reluctant about providing financial resources for IT. In a remote past the used to say: “There is no money for IT. Use hands to make calculations!” Obviously this not makes sense today but this feeling is met in many decisions everywhere.

This paper has the purpose of blink a yellow signal to managers, who have the role of making important decisions in organizations, to take attentions to the following aspects:

- When we think about quality in product manufacturing, we have to think in a joined action of Manufacturing Process and IT Resources which are essentials to reach the goals of the organization;
- Global directives are always well come but we have to attempt to the fact that these directives do not promote difficulties to the branches. It is recommended to analyze the resulting costs compared with the benefits obtained. Adaptations or directives variants could be possible solutions to attend to the global directives and at the same time minimize the negative effects. All should be balanced before make this changes;
- IT area requires substantial financial resources. The not enough investment in IT starts a degradation process whose recovery in the future could be much more onerous;
- It is necessary to face IT as a co-worker using it in a good way, according to a well-designed corporative governance. The technology in the IT area in terms of infra structure (Hardware) or in terms of numeric process or algorithms (Software) are both available and could be applied in many purposes and subjects of great interest to the organizations such as:
  - Reduction of cost in general;
  - Production Performance Increment;
  - Human Resources Allocation Optimization;
  - Logistic;
  - Quality Improvement;
  - Planning;
  - Marketing;
  - Etc.

The good planning suggests the investment of an adequate amount of financial resources in IT area, to make possible: the quality goals reaching; exceeding the customer
expectations; making quality improvement using IT resources; inserting digital technology in the product manufacturing and control; using techniques of optimization in cost reduction, manufacture performance improvement, logistic, maintenance, etc.

There are many sectors of the company where IT can actuate and the way of doing that are also numerous and promising. The investment in IT contributes to the organization growing. It is enough for that to understand that the product manufacture and IT walk side by side and the sum of both efforts produce important benefits assuring the organizations competitive, through products and services with quality, which are essential for the global world we are living.

REFERENCES


CONSULTED REFERENCES


ABOUT THE AUTHOR

The author is an Engineer from São Paulo University with 30 years of experience and has been working at Delphi Diesel do Brasil for more than 10 years as independent consultant. In the University, as a researcher, he is finishing the PhD program this year, making researches about solutions of complex problems related to optimization in several areas such as logistic, power generation, just in time process, etc.

Contact could be made by: lelistm@lechare.com.br or cell phone +55 11 9626 6736 and through URL www.lechare.com.br in contact page.