RESEARCH ARTICLE OPEN ACCESS

OQM Model: A Case Study of Software Quality

Mrs. Neha Agrawal
Banasthali University
Prof. (Dr.) Praveen Dhyani
Executive Director, Jaipur Campus
Banasthali University

ABSTRACT

Performance evaluation of software quality delves into identification of metrics required at various categories in the value chain of software industry. Various performance models and software quality concepts have been evaluated and critically appraised. Gaps in the literature point to the need for a more realistic business empowered measurement system for software quality. In the last few years the software industry has witnessed rapid growth and has experienced several innovations.

Keywords:- QMM, DSS, KPI, EO, NGT

I. INTRODUCTION

There is a greater emphasis on measurements for effective decision making. Empowered by measurement, the modern decision maker is able to free himself from prejudice and move towards objectivity. Lack of visibility is a well known constraint in software project management. Software measurements bring visibility into process of software development, management, installation, maintenance and use. Measurement constitutes the foundation of a new culture. The process of measurement establishes an environment of observation and opens closed minds.

Three phases of measurement co-exist. There is a cognitive phase where measurement begins with perception and all constituents of mind are at play. There is a semantic phase where semantic expressions are used to label or refer to the observation which is known in measurement science as a nominal scale. There is a quantitative phase where numbers are used to indicate value, to represent quantities and to donate levels. Quantitative phase permits construction of mathematical equations and advanced analysis. These phases are not to operate in isolation. There exits an evitable plurality in measurement methods. From this perspective, numbers are extensions of an existing system of observations, thinking and communication.

The Objective Question Metrics (OQM) Model proposed in this work explores the various refined business objectives. It provides a multi-tiered approach to quality managers and metrics analysts to pick and choose a wide array of metrics as per their needs and choices based on enterprise objectives rather than goals which earlier models prescribed. OQM Model of teir-1 focuses on first level enterprise objectives but as companies start identifying newer objectives but as companies start identifying newer objectives based on their business performance, multi-tier OQM Models may emerge. OQM Model provides online data collection mechanisms powered by multiple enterprise wide tools. OQM DSS is based on a metric engine of 47 metrics which are filtered at two stages and thus enable the management to leverage the metric intelligence for their decision making based on data and facts.

Unlike other engineering disciplines, Software Engineering is not grounded in the basic quantitative laws of Physics. Measures such as voltage, mass, velocity or temperature are uncommon in software model. Instead we attempt to derive a set of indirect measures that lead to metrics, that provide indication of the quality of representation of software as Software metrics and measures are not absolute and open to debate. Confusion prevails in attaining distinction between measures, metrics and measurement.

Software Productivity poses further more challenge as one starts calculating programmer productivity. Jacob (2005) continues to argue that size of the code alone does not contributed to software productivity. A Perl Code can be written to be small in size but difficult to read and thus could consume more time of programmer. He cites that if a bunch of programmers are locked in different rooms with the same specification and each programmer uses a different language or paradigm it would be difficult to compute productivity of the programmer.

Measures provide quantitative indication of extent, amount, dimension, capacity or size. Measurement occurs as a result of collection of data points or measures. Software metrics as defined in IEEE standard refers to quantitative measure of degree to which system, component or process possess a given attribute. Moving from measurement to metrics is like moving from observation to understanding. Several rules have been prescribed to plan metrics and metrics are best viewed as systems. One cannot design metrics in isolation from environment. The metric developed as the part of study discusses about the metric system built around the information highway of organization. The objective of the metric development has been primarily to provide model based decision support. It is seen through the literature survey and survey of quality measurements that many measures or metric have emerged by focusing on project management goals and software development goals. Craig (2002) has discussed and brought out clearly the distinction between goals and objectives. Objectives on the other hand, are specific and measurable. Think of the word "go". It has no end. Goal comes from "go" and think of the word "object". Object can be touched, it is actual and finite. Department of Energy promulgated a set of Total Quality Management guidelines that indicate that performance metrics should lead to quantitative assessment of gains in Customer Satisfaction, Organizational Performance and Work force excellence. However in the present research work, Nominal Group Technique (NGT) has been adopted to find the best methodology to collect views to propose a model for Software Quality. Probing Questions for Development of Enterprise Objectives (EO)

Legend: Brackets indicate Mapping of Questions to Tier-1 OQM Metric Set

The choice of KPI of a person would vary year to year based on Strategic Planning, Corporate Strategies or Corporate Strategies or Corporate/Enterprise Objectives Metric Development through NGT Method from Gamma Stakeholders

Goals	Questions	Metrics
G1: Improve Development Process	Q11: How well does the development process describe the work being performed? Q12: What is the relative effort for each activity in	M11: Average elapsed time between defect identification and correction. M12: Number of person hours (effort) to complete each activity. M13: Elapsed time for each

	the process? Q13: What is the elapsed time for each activity in the process? Q14: Where in the process are defects being introduced? Detected? Corrected? Q15: How many requirements are added during the process?	activity. M14: Number of defects detected in each activity. M15: Number of deviations from the software development process M16: Number of requirements added or changed during development
G2: Improve Software Estimation	Q21: What is the actual versus estimated labor rate for each activity? Q22: How much have the requirements changed since the initial estimates were made? Q23: How complicated is the software being developed? Q24: What is the actual versus estimated schedule, effort, and size for each activity? Q25: What is the actual versus estimated staffing level? Overtime worked?	M21: Initial estimate versus actual effort (person hours) for each activity. M22: Initial estimate versus actual project schedule for each activity. M23: Initial estimate versus actual size of the software (new and reused). M24: Initial estimate of staff required versus actual staff levels (for each activity) M25: Total overtime hours M26: Labor rate (PH/SLOC) for each activity. M27: Requirement changed for each activity M28: Software product complexity.
G3: Improve Project Tracking	Q31: What is the status of each development activity Q32: what is the status of overall Project Q33: What is the earned value of each activity? Q34: How do actual project expenditure	M31: earned value of each activity M32: SLOC Completed/Schedule Variance M33: Initial estimate of SLOC. M34: Overall percent of work complete M35: Percentage of work complete for each activity M36: Percentage of budget spent up to date.
G4: Minimize Development Cost	Q41: What is the cost of each activity? Q42: What is the labor rate of each activity? Q43: What is the original versus actual effort required for each activity?	M41: Actual cost of each activity. M42 Amount spent on fixing defects M43: Initial cost estimate of each activity. M44: Budget for each activity. M45: Initial effort versus actual

	Q44: How much of the budget is spent on development versus managerial versus support task? Q45: How much of the budget is spent to correct defects?	effort for each activity M46 Percentage of budget spent on development/management/support tasks
G5: Improve Software Quality	Q51: How many defects are there in the product? Q52: Is the software maintainable? Q53: Has the software been verified? Is it correct?	M51: Average person hours to fix a defect M52 Mean time between failures (if appropriate M53: Number of defects detected of each type. M54: Number of defects/SLOC M55: Percent of code inspected.
G6: Improve Software Performance	Q61: What is the processor utilization? Q62: What is the memory utilization? Q63: How is the software I/O performance? What are the characteristics of the software?	M61: Average CPU utilization M62: Average memory utilization M63: Mean time between failures (if appropriate) M64: Number of I/O transactions per unit of time (actual versus required). M65: Number of lines of code (SLOC) M66: Software product complexity
G7: Improve Software Productivity	Q71: How much time is being spent on rework? Q72: Are developers spending too much time on support and managerial activities? Q73: What is the average productivity? Q74: Is the productivity consistent with the experience of the team members? Q75: Are tools available to use to answer these questions about productivity	M71: Average number of person's hours spent on rework per development staff member. M72: SLOC/person hours for each activity. M73: Number of staff at each experience level. M74: Percent of budget available for software development tools. M75: Percent of budget available for support staff. M76: Proportion of person hours spent on managerial or support tasks for each activity. M77: Ratio of development staff per manager.
G8: Minimize Schedule Overrun	Q81: What is the actual schedule of the activity? Q82: What is the actual	M81: Initial estimate v/s actual estimate. M82: Initial schedule v/s actual

	level of effort and rework? Q83: Is staffing level adequate to meet schedules?	schedule M83: Initial estimate v/s actual staffing levels M84: Staffing Variance

On further investigation, it has been found that there are 47 metrics coming out of this framework initiative. The Metric and their explanation have been listed in Table 4

Metric listing and Explanation

SI no.	Metric	Description	Explanation
1	M11	Average elapsed time between defect identification and correction.	Defect resolution time indicates the time elapsed between identification of defect and resolving them. This time is critical Metric as defect aging needs to be minimized.
SI no.	Metric	Description	Explanation
2	M12	Number of person hours (effort) to complete each activity	Number of Person hours indicates effort to complete each activity. It could be in man months, man-hours, person-days or person-days. It is computed by multiplying time with number of persons.
3	M13	Elapsed time for each activity	Effort distribution time indicates how much of time is consumed for each stage of software development-Effort distribution time.
4	M14	Number of defects detected in activity	Defects in each stage of life cycle.
5	M15	Number of deviations from the software development process	Process Nonconformance as per mandated Life cycle model like Waterfall, RAD, Spiral and Iterative Prototyping.
6	M16	Number of requirements added or changed during development.	Variance in TCSER (Time, cost, Size, Effort and Resources).
7	M21	Initial Estimate versus actual effort (person hours)for each activity	Effort Variance
8	M22	Initial Vs. Actual Project Schedule for each activity	Schedule Variance
9	M23	Initial Estimate Vs. Actual Size of Software (for each activity	Size Variance
10	M24	Initial Estimate of Staff required versus actual staff levels.	Staffing Variance
11	M25	Total Overtime Hours	Schedule Variance
12	M27	Requirements changed for each activity	Scope Creep Index/requirement Volatility

13	M28	Software Product Complexity	Complexity Measure of Mc Cabe
14	M31	Earned Value of each activity	Earned Value Management
15	M32	SLOC Completed	Work throughput
16	M33	Initial estimate of SLOC	Unit of Effort Estimate
17	M34	Overall percentage work done	Work distribution
18	M35	Percentage of work complete for each activity	Work accomplishment pattern
19	M36	Percentage of budget spent up to date.	Budget Usage pattern
20	M41	Actual cost of each activity	Cost of operation
21	M42	Amount spent fixing defects in each activity	Defect Resolution Cost
22	M43	Initial Cost of Estimate of each activity	Activity Based Costing
23	M44	Budget for each activity	Budget Allocation
24.	M45	Initial Effort Vs. actual effort for each activity	Effort Variance
25	M46	Percentage of budget spent on development/management/support tasks	Engineering Budget/Support Budget/Operations cost
26	M51	Average Person hours to fix defect	Defect Resolution Rate (Average)
27	M52	Mean Time between Failures	Reliability Measure
28	M53	Number of defects detected in each type	Defect Distribution (Serverity Levels)
29	M54	Number of defects/SLOC	Defect Density Measure
30	M55	Percent of Code Inspected	Code inspection coverage
31	M61	Average CPU Utilization	Resource Utilization Measure
32	M62	Average Memory Utilization	Resource Utilization Measure
33	M63	Mean time between failures	Reliability Measure
34	M64	Number of I/O Transactions per unit of time	I/O Distribution
35	M65	Number of Lines of Code (SLOC)	Effort Indicator
36	M66	Software Product Complexity	Complexity methods
37	M71	Average number of person hours spend on rework per development staff member	Rework percentage per developer

38	M72	SLOC/Person hours for each activity	Effort indicators (work Vs. time ()
39	M73	Number of staff at each experience level	Experience Level Distribution
40	M74	Percent of Budget available for Software development tools.	Budget Allocation for Tools
41	M75	Percent of Budget Available for support staff	Support Staff Budget Allocation
42	M76	Proportion of person hours spent on managerial or support tasks for each activity	Effort Distribution (Support)
43	M77	Ratio of development staff per manager.	Span of Control measure
44	M81	Initial Estimate vs. actual Estimate	Estimation Effectiveness
45	M82	Initial Schedule vs. Actual Schedule (effort and rework0	Schedule Overrun (effort and Rework)
46	M83	Initial Estimate Vs. Actual Staffing Levels	Resource Overrun
47	M84	Resource Leveling for schedule Variance minimization	Staffing Level Variance

II. SELECTION OF METRIC

This section discusses how to shortlist 47 metrics and relate the same to Enterprise Objectives and Critical Success Factors for three levels of Software Organizations operating in the Value Chain. Organization who wants to measure his/her Organizations objectives. Secondly Enterprise Objectives of the organization and Critical success factors of Software Company need to be considered before arriving at a final metric set.

Decision making in selection of metrics for the OQM Model is carried out at two stages.

- Stage 1 Filtering: Eight Goals arrived from NGT technique by polling primary and secondary stakeholders have been pitted against 6 Enterprise objectives to validate the mapping strength and relationship.
- Stage 2 Filtering: Stage 1 resulted in 5 goals and 27 metrics which is against pitted against three levels of software organizations (in Software Value Chain of Indian IT Industry)

In stage 2 the primary focus is on Software Value Chain and mapping of Critical Success Factors (CSF s) of the Value Chain. Multiple Criteria like 8 Goals, 6 Enterprise Objectives and Software Value Chain has been taken into consideration for taking a decision on selecting the most significant metric set which is closer to Corporate Objectives. MCDM yields different results when Enterprise Objectives changes based on business outcomes and business performance.

Filtering provides dynamicity in the OQM model as multiple permutations and combinations of metric set can be generated out of 47 metric set based on Organizational/Enterprise Objectives.

Procedure for Level-1 Screening

- Step 1: Eight Goals derived from were placed in Rows
- Step 2: Weights were assigned on a scale of (1-10) for each of the cell based on the assumptions and market trends and independent of any Software Organization categories
- Step 4: Row weights were normalized by dividing individual row weights by Mean Column weights of Row Weights column.
- Step 5: Top five values are selected are G1, G3, G5, G7, and G8
- Step 6 : As per Table each of these goals provides 27 Intermediate Metrics

Procedure for Level -2 Screening

- Step 1: Intermediate Metric set of 27 metrics Step 2: resulting metrics were mapped against three levels of Software Organizations in the value chain. Table 4.9 discusses the same. In this Table the CSP of each level and the metric is mapped by assigning weights or significance to see how far does the intermediate metric address the CSP of the Software Organization Level or Category.
- Step 3: It is found that M11 which corresponds to response time shows high affinity relationship with Level-1 Organizations, M51 which corresponds to product defects shown high affinity relationship to Level-3 Software Organizations, M74 and M75 which corresponds to training and tools to ensure employee satisfaction shows high affinity invariably to all Levels of Software Organizations and M81 which corresponds to schedule variance shows high affinity relationship to Level-2 organizations.
- Step 4: Hence M11, M51, M74, M75, M81 together with EO2 and EO3 enters the Tier-1 OQM Model for Software Organizations. Step 4 of Section 4.5.1 discusses how the normalized weights have been arrived at.

III. PERFORMANCE EVALUATING USING OQM MODEL

MO11 Schedule Variance: This metric is a result of significance attached to Level-1 Software Organizations which is based on billing models like Time and Material, Fixed Price or Fixed Time model. This metric is a critical factor to achieve Business Success and is a progress indicator. This is a project Management metric where for any projects executed By Level-2 Software Organizations the model warrants tracking of Start Variance and End Variance. End Variance is (Var Days/Act Day)* 100 and Start Variance is (Var Days/Plan days)*100. Var Day = (Act Days-Plan Days). Project Management Body of Knowledge (PMBOK) defines Schedule variance during the start and end of the project. Start variance means when a project manager starts a project, he or she would like to know how much delay was there in staring the project against plan start date. Similarly End Variance indicates the delay of the project i.e. the gap between the actual end date and planned end date.

MO$_{12}$ and MO$_{13}$ Product Field Defects and Severity Field Defects: This metrics hold importance for organizations in Level-3 making software products and are at very high level in the software Value Chain. Here the number of open defects product wise and severity wise is captured and attempts are made to control the defects to achieve Product Quality.

MO$_{14}$ Turnaround Time: This the resolution time or the time elapsed between raising a defect or complaint and closing the same. It is found that in Level-1 Customer Care organizations like BPO/Contact Center this metric holds significance and is a CSP for Level-1 organizations apart from metrics like number of calls made by agent per hour, number of contacts per hour or it could be conversion rate of calls to checks in case of debt collection process in a BPO.

MO$_{15}$ and MO$_{16}$: Employee Satisfaction Index and Customer Satisfaction Index CSI/ESI) this metric discusses the need for a measure where all the internal and external customers are happy and satisfied and giving their best for the growth and excellence of their organizations. This metric holds high significance to all types of software Organizations in the Value chain.

IV. CONCLUSION

The basis of preparing the Tier-1 OQM Model and subsequently Tier-n too with permutations among 47 metrics given a new enterprise objective. The metric list would vary in tier-2 when the software category is different from the categories taken in OQM Framework. Having defined the OQM Framework, in the succeeding chapter the applications of OQM Model to different levels of Software Industry is mapped to see what are the cores or critical success factors in each of Software Industry and how OQM could be leveraged to address the same. The proposed OQM Model which resulted in 6 metrics in Tier-1 filtering when go applied to the case companies proved to be a empirical fit as it was found during the validation process for Alpha technologies at Level-3 resulted in two metrics on Product field defect and Field defects severity wise. Similarly while validating the model for Gamma Technologies resulted in turn Around Time as critical success factor. Major validation of the model was carried out by conducting NGT with the CEOs of Alpha, Beta and Gamma Technologies.

Case study method details contextual analysis of a limited number of researcher and in general Social scientists has used the Qualitative research method as it is used in the development of OQM Model. In Qualitative research wide use of experience, real-life situations form the basis for application of ideas.

For the development of OQM Model the researcher had adopted the above steps of Case based method. The research object was Software Quality and an exploratory research was conducted to see whether there existed a model which can evaluate whether there existed a Model which can evaluate an organization for Business Performance and Effectiveness. The candidate companies selected were from Indian Software Industry and three levels of organizations in proposed software were from Indian Software Industry and three levels of organizations in proposed software value chain has been taken. Data was collected for Tier-1 OQM Model and evaluation and analysis was done and the scheme for collection of data for multi-tier was also proposed. The companies were selected based on convenience sampling. The researcher selected the Gamma technologies, Alpha and Beta Technologies close to his place and study. This ensured more interaction, observation and facilitated more idea engineering and brainstorming from the respondents during data

On the lines of manufacturing value chain propounded by Poter (1980) attempt has been made in the research work to map software value chain with primary activities and secondary activities. Engineering Process like requirement Management, Design, coding, Unit Testing and User Acceptance testing as per SDLC Models addresses primary activities. To ensure the software is of good quality and delivered on time, cost, plan and resources feeder processes like quality Management Processes, Project Management Process is considered as engineering activities.

REFERENCES

[1] Abreu, FB., "Metrics for Object-Oriented Environment," Proceedings of the Third International Conference on Software Quality, Lake Tahoe, Nevada, 1993, pp. 67-75.

[2] Agile Alliance At http://agilealliancebeta.org/article/file /904/file.pdf, 2001

[3] Akao, Y., "New Product Development and Quality Assurance system of QFD Standardization and Quality Control", Japan Standards Association, Vol 25(4) 1997, pp 9-14.

[4] Albrecht, A. J., "Measuring Application Development Productivity," Proceedings of the Joint IBM/SHARE/GUIDE Application Development Symposium", 1987, pp. 83-99

[5] Development Effort Prediction: A Software Science Validation", IEEE Transactions of Software. Engineering Vol. 9 (6), 1983, pp. 639-648.

[6] Arnold, R, S. "A Road Map Guide to Software Reengineering Technology", Software Reengineering. IEEE Computer Society, 1992, pp. 3-22.

[7] Azuma, M., "Software Quality Assurance", Vortrags manuscript zum Vortrag, Vol 12(6), Zurich, Germany, 1987.

[8] Bailey, C.T., and Dingee, W.L., "A Software Study Using Halstead Metrics", Bell Laboratories Denver, CO. 80234, 1981.

[9] Baker, A.L., Beman, J.M., Gustafson, D.A., Melton, A., and Whitty, R.A., "Modeling and Measuring the Software Development Process", Proceedings of the Twentieth Annual International

[10] Baker, A.L., Bieman, J.M., Fenton, N., Gustafson, D.A., Melton, A. and Whitty, R.A., "A Philosophy for Software Measurement", The Journal for Systems and Software, Vol. (3), 1990, pp. 277-281

[11] Barns D. ,and Mechael, G., "Inheriting Software Metrics", Journal of Object Oriented Programming, 1993, pp.27-34.

[12] Basili, V.R., Reiter, R., "Evaluating Automatable Measures of Software Development", Proceedings of Workshop on Quantitative Software Models, 1989, pp. 107-116.

[13] Basili, V., and Rombach D.H., "Integrating Measurement into Software Environments", TR-1764; TAME-TR-1, 1987.

[14] Bazzana, G., Anderson, O., and Jokela, T., "ISO 9126 and ISO 9000: Friends or foes"? Presented at Software Engineering Standards Symposium, 1993.

[15] Belady, L.A., On Software Complexity i: Workshop on Quantitative Software Models for Reliability. IEEE No. TH0067-9, New York, N.Y., pp 90-94, 1979.

[16] Beck, Ke., "Extreme Programming Explained: Embrace Change", Boston, MA: Addison-Wesley, ISBN 0-321-27865-8", 2001.

[17] Biehl, R.E., "Six Sigma for Software", IEEE Software, Vol. 21(2), pp 68-70, 2001

[18] Bieman, J.M., and Schultz, J. "An Empirical Evaluation and Specification Testing Criterion", Software Engineering Journal, Vol. 7(1), 1992, pp 43-51.

[19] Bieman, JM., "Deriving Measures of software Reuse in Object Oriented Systems", Technical Report #CS91-112, Colorado State University, Fort Collins/Colorado, USA, 1991.

[20] Boddie, J. , "Do We Ever Really Scale Down?", IEEE Software, Vol. 17(5), pp. 79-81, 2000.

[21] Boehm, B. W., Brown, J. R., Kaspar, J. R., Lipow, M. L. & MacCleod, G., " Characteristics of Software Quality", New York: Americal Elsevier, 1988.