MERIT RESEARCH JOURNALS
www.meritresearchjournals.org

Merit Research Journal of Business and Management (ISSN: 2408-7041) Vol. 4(3) pp. 018-026, June, 2016
Available online http://www.meritresearchjournals.org/bm/index.htm

Review

Recent Advances in Software Quality Management: A Review

Boby John[1*], R. S. Kadadevaramath[2] and Edinbarough A. Immanuel[3]

[1]Indian Statistical Institute, Bangalore, India

[2]Siddaganga Institute of Technology, Tumkur, India

[3]The University of Texas at Brownsville, Texas, USA

***Corresponding Author's Email: boby@isibang.ac.in**

Abstract

Many organizations has been utilizing the benefits of information technology to gain competitive advantage in their respective businesses. As a result, the number of software development companies increased many folds during the last three decades. As the competition increased, the need for delivering good quality software within the committed schedule also increased. Even today many software companies have to deal with the consequences of delivering poor quality products, schedule and cost overrun problems. Many software quality and development frameworks have been suggested in the past to get rid of the aforementioned problems and a lot of research has been carried out in the field of software quality management and practices. In this paper, the authors provide a review of the major research works published in the field of software quality management. The study found that the research works in the software quality can be classified into five categories namely studies exploring the relationship between software quality and (i) total quality management implementation, (ii) adoption of quality management systems like ISO 9000 series of standards, (iii) capability maturity model level, (iv) development of defect prediction models and (v) quality management and evaluation practices of specific categories of software development process. The authors also tried to identify the gaps in existing research and future areas of research in software quality management.

Keywords: Capability maturity model, Defect prediction models, ISO 9000, Quantitative project management, Software quality, Total quality management

INTRODUCTION

Surviving in a globalised business world is not easy unless the organizations have a competitive advantage (Kanji and Asher, 1999; Samason and Terziovski, 1999 and Adam et al., 2001). During the past three decades, many organizations deploy information technology (IT) to gain the aforementioned competitive advantage. The IT industry has witnessed tremendous growth in the last three decades. The Gartner group estimated even a decade ago that the worldwide user spending on software exceeded $730 billion and that in the packaged software market topped $176 billion (Shiffler, 2003; IDC, 2003). Many business organizations are using IT as a means to achieve operational efficiency, improved productivity and service quality and responsiveness (Mooney et al., 1996).

As the demand for software products increased, the number of software development firms also increased. Even though the number of software development firms increased rapidly, delivering quality software products without cost and schedule overrun has been always a challenge in software industry (Pearson et al., 1995; Phan et al., 1995). A study by US General Accounting Office (1973) reported that many government software projects were never delivered or couldn't be used and had cost over runs or schedule overruns. The breakup of software project problems (Osmundson et al., 2002)

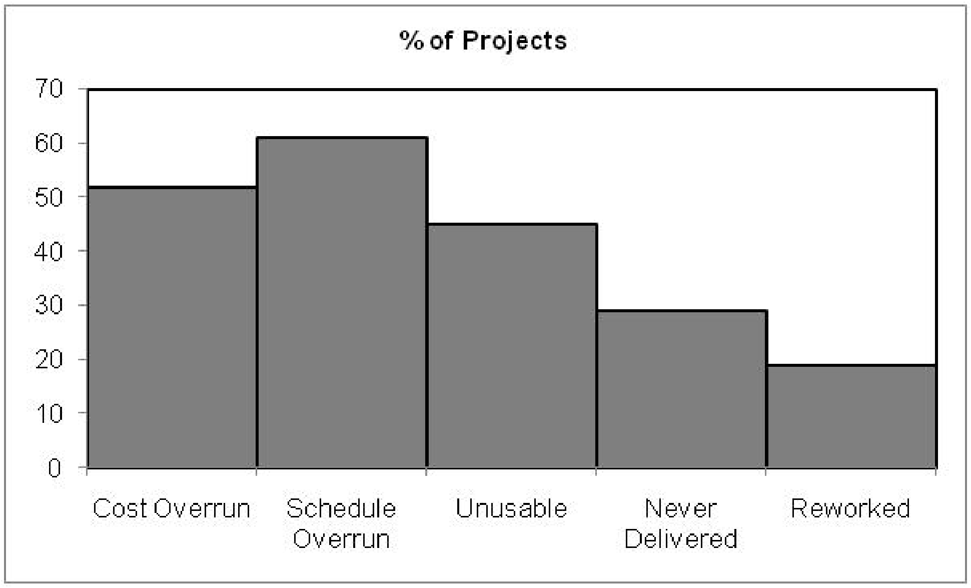

Figure 1. Break up software projects with problems

published by the study is given in figure 1.

Many software firms has comprised on quality to meet the schedule pressure and control the development costs leading to detecting higher number of defects at customer end (Kemerer, 1997). The lack of quality had significant costs to the suppliers who face dissatisfied customers, loss of market share and rework of rejected systems (Sibisi and Waveren, 2007). Hence producing quality software within the committed time and cost are very important for the survival of software development organizations in the highly competitive information technology industry.

Defining software quality is not easy and there is no single adequate measure for software quality. Fenton and Bieman (2014) suggested measuring software quality in terms of delivered defect density. The delivered defect density is the number of defects per unit size. According to ISO/IEC 9126 – 1, the software quality model has six characteristics namely functionality, reliability, usability, efficiency, maintainability and portability (Al-Kilidar et al., 2005). ISO 9126 (1991) defined software quality as the totality of features and characteristics of a software product that bear on its ability to satisfy stated and implied needs of the customer. The widely adopted Capability Maturity Model (CMM) developed by Software Engineering Institute (SEI) of the Carnegie Mellon University classifies the software process into five maturity levels namely initial, repeatable, defined, managed and optimizing (Paulik et al., 1994). In short, there has been a variety of approaches and guidelines suggested for software quality management and improvement.

The purpose of this paper is to review the literature on recent advances in software quality management and identify the major gaps in the research. The remaining part of the paper is organized as follows: The review methodology is described in section 2, the literature review analysis is given in section 3 and the conclusions and areas of future research are summarised in section 4.

REVIEW METHODOLOGY

In the recent past numerous books have been written, papers have been published, conferences have been organised on various approaches for measuring and improving software quality and software quality management practices. The aim of this paper is to provide a comprehensive review on the recently published scientific literature on software quality management and related topics.

The review process started with searching for relevant articles. The focus is on papers and books published in English from 1990 onwards. Only articles published in journals, books or presented in conferences are included in the review. The authors searched the popular research databases like ScienceDirect, ieee xplore, google scholar and researchgate using software quality, software quality management or software quality prediction as key words.

Six hundred and twenty four articles suggested by the databases are identified for the survey. After going through the key words and abstracts, 196 articles were shortlisted for detailed study. Another 79 papers were also excluded after reading the full text as they were not directly related to the topics of interest. The remaining 117 articles were reviewed in this study. The details are given in figure 2.

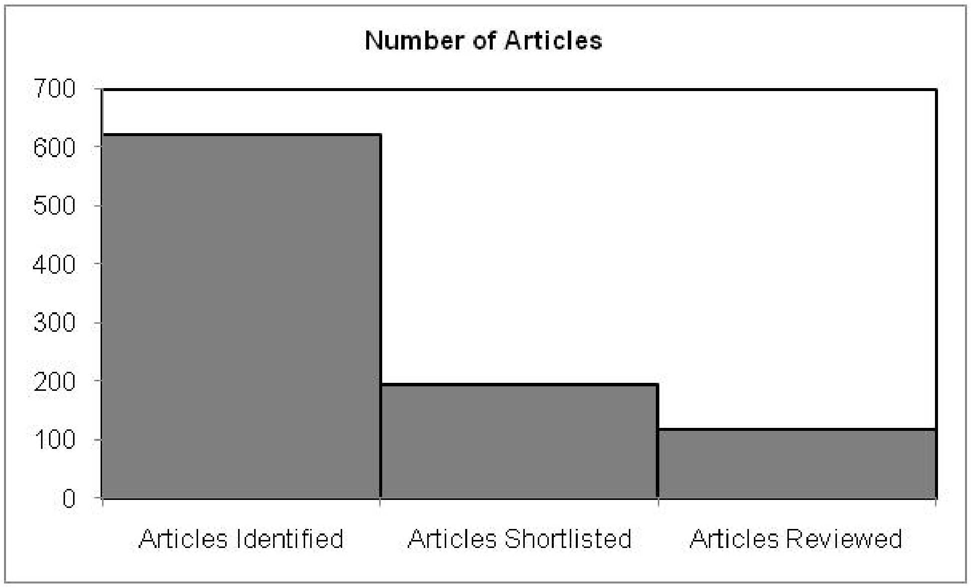

Figure 2. Details of articles chosen for review

Table 1. Category wise number of articles reviewed

Number	Category	Number of Articles
1	Total Quality Management	13
2	Quality Management Systems	10
3	Capability Maturity Model	11
4	Quality Prediction Models	70
5	Others	13

The details of review analysis for each groups is as follows:

Literature Review Analysis

The authors carried out detailed analysis on the articles selected for the study and classified the articles into five different categories based on the topics of their research. The grouping had enabled the authors to summarise the findings and identify the future research areas. The number of papers reviewed under each category is given in table 1.

Total Quality Management and Software Quality

The universal acceptance of the TQM has inspired software firms to adopt it to their industry (Cortada, 1995, Alkhafaji et al., 1998; Bhattacharya et al., 1998). Parzinger and Nath (2000) studied the relationship between TQM implementation factors and various measures of software quality. The study showed that the TQM implementation factors have significant positive correlation (p value < 0.01) with the software quality measures except cost of quality (p value > 0.05).

Li et al., (2000) suggested an approach to instil TQM method in software development process. Issac et al., (2004) proposed a holistic conceptual framework for implementation of TQM in software industry. The framework suggest to constantly measure the performance of the system using metrics, analyse with respect to the benchmarks set and provide feedbacks to the system to take necessary corrective steps. Carrol (1995; Camuoff et al., 1990) argued that the application of the key elements of TQM in software development has the potential to improve the software quality. Many researchers (Munson and Khoshgoftaar, 1992; Zardony and Tumanic, 1992; Gong et al., 1998) studied the application of TQM concepts like statistical quality control and quality function deployment on software quality. Walrad and Moss (1993) observed that the impact of TQM techniques on system quality depends on effective linking of product and process metrics to system quality objectives.

Even though the research suggested positive correlation between TQM implementation and software quality, most of the works were on conceptual framework or empirical studies based on the analysis of survey data.

Quality Management Systems and Software Quality

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) has published two series of standards namely ISO/IEC 9126 for software product quality and ISO/IEC 14598 for evaluation of software products (Suryn et al., 2003). The ISO/IEC 9126 standard suggested a quality model comprising of six characteristics and 27 sub characteristics of software product quality.

Shem et al, (2015) showed that generally there exists positive correlation between implementation of standards and product quality. Jung et al, (2004) conducted a survey among users of a packaged software product to evaluate the structure of the software quality model proposed in ISO /IEC 9126 – 1. The study revealed that ambiguities exist in the way that ISO/IEC 9126 is structured and the sub characteristics categorization isn't consistent with the ISO/IEC 9126 definition. Balla et al., (2001) published a case study on the success of ISO certification in a software company in Hungary. According to the study, working in conformance with QMS caused about 30% of extra effort in small projects and about 10 – 20 % in bigger projects. But the advantages were perceived to outweigh the costs. Yang (2001) studied the attributes of the software quality given in ISO 9000 standard and its usefulness in estimating the software product quality. Paulk (1993) observed that there exists strong correlation between ISO 9000 standard and CMM frame work. Franch and Carvallo (2003) developed a quality model for choosing off the shelf software packages based on ISO / IEC 9126 – 1 standard. Coallier (1994) explored the fitment of ISO 9000 standards for software development process. The study found that ISO 9001 / 9000 – 3 only partially supports the assets needed to deliver the software when it is required with minimum life cycle costs. Al – Kilidar et al., (2005) reported an empirical study on application of ISO / IEC 9126 to software design documents and found that the standard has ambiguity and overlap in some concept definitions, doesn't consider reliability and maintainability as well as validity and modularity of design products. Jenner (1995) showed how ISO 9001 can be used for software development process.

The review of the literature had thrown out conflicting findings. While many of the studies suggested improvement in quality with the implementation of quality management systems or standards, there were studies disagreeing with the aforementioned conclusion. Unfortunately most of the papers were empirical studies based on survey data.

Capability Maturity Model and Software Quality

The capability maturity model (CMM) developed by the software engineering institute (SEI) of Carnegie Mellon University is one of the widely accepted framework for characterising the capability of software development processes (Pressman, 2005).

Harter et al., (2000) empirically investigated the relationship between process maturity measured in CMM maturity scale, product quality, development cycle time and effort. The study found that 1% improvement in process maturity is associated with a 1.589% increase in product quality. The other findings of the study are higher the quality, lower is the cycle time and development effort. Herbsleb et al., (1997) also showed that CMM levels influence the software quality and project performance variables. Zimmerman (2001) suggested that poor planning and lack of training are two root causes of software project failure. Many other studies also showed that higher CMM levels are associated with improved software quality (Zimmerman, 2001; Lawlis et al., 1995; Krishnan, 1996; Subramanian et al., 2007).

But Hansen et al., (2004) pointed out that the evidence for higher CMM levels are associated with higher quality are limited and not based on reflective models.

Higher maturity levels (level 4 and 5) of capability maturity model requires the quantitative management of the process (C P Team, 2006), which means the process need to be controlled by statistical and other quantitative techniques. Tamura (2009) developed three process performance models using regression techniques, one for achieving product quality objectives (defect density) by controlling requirement inspection rate (pages per hour) and prototype developed or not. The second one for achieving code review yield (% of defects present in the software that are removed by the review) targets by controlling review rate (the number of lines of code reviewed per hour). The third model is to manage the escaped unit test defect density using test coverage as controllable variable. Hao and Zhang (2011) also developed a model for delivered defect density in terms of average team skill level and test coverage as controllable variables.

Majority of the papers suggested that higher maturity levels were attached with better software quality. Except the research on developing process performance models, the remaining articles were mostly based on empirical evidence.

Development of Defect Prediction Models

From 1970s onwards one major area of research in software quality has been the development of defect prediction or classification models. The important papers published on defect prediction models from 1990s onwards along with the methodology used is summarised in table 2.

The breakup of different techniques used to develop the prediction models is given in figure 3. Most of the models used static code attributes like code complexity, etc as predictors those are not routinely measured by the project managers and may not be under their control.

Table 2. Research on developing defect prediction models

SL No	Methodology	Details
1.	Regression Analysis	Regression with different intercepts and slopes (Nagappan et al, 2005), Logistic regression (Khoshgoftaar and Allen 1999; Cruz and Ochimizu 2009), Zero-inflated negative binomial regression (Succi et al., 2001), Zero – inflated Poisson regression (Khoshgoftaar, 2001), Regression via classification (Bibi et al., 2006), Linear mixed-effects regression models (Binkley, 2007)
2	Multivariate Analysis	Factor analysis (Khoshgoftaar and Munson, 1990; Munson and Khoshgoftaar, 1990), Principal component analysis (Niel, 1992; John and Kadadevaramath, 2014), Discriminant analysis (Schneidewind, 2001; Khoshgoftaar and Seliya, 2002)), Cluster Analysis (Sandhu et al., 2010), Singular value decomposition (Sherriff et al., 2007)
3	Machine learning techniques	Neural Networks (Khoshgoftaar et al., 1995; Khoshgoftaar et al., 1997; Mair et al., 2000; Kanmani et al., 2004; Pedberg et al., 2004; Thwin, and Quah, 2005; Bezerra et al., 2007; Kanmani et al., 2007; Singh et al., 2008;Tao and Wei-hua, 2010), Bayesian Networks (Fenton and Niel, 1999; Amazaki et al., 2003; Fenton et al., 2007, Menzies et al., 2007,Pai and Dugan, 2007; Turhan and Bener, 2007; John, 2012), Classification and Regression Tree (Porter and Selby, 1990; Khoshgoftaar et al., 1999; Koru and Liu, 2005; Menzies et al., 2003; Khoshgoftaar and Seliya, 2003b; Khoshgoftaar and Seliya, 2002; Knab et al., 2006; Ceylan et al., 2006), Random Forest (Guo et al., 2004; Kaur and Malhotra, 2008), Rough Sets (Morasca and Ruhi, 1996; Yang and Li, 2008), Instance based learning (Ganesan et al., 2000; Emam et al., 2001; Khoshgoftaar and Seliya, 2003a; Khoshgoftaar et al.,2006; Challagulla et al., 2006), Support Vector Machines (Elish and Elish, 2008), Self Organizing Maps (Reformat et al., 2003; Mahaweerawat et al., 2007), Genetic Programming (Afzal and Torkar, 2008; Afzal et al., 2008), Particle Swarm Optimization (De Carvalho et al., 2008; Cong et al., 2010; De Carvalho et al., 2010), Grey Prediction Theory (Zhu and Wu, 2009), Dempster-Shafer Belief Networks (Guo et al., 2003), Fuzzy Logic (Xu et al., 2000; Yuan et al., 2000; Reformat, 2003; Yang et al., 2007; Hribar and Duka, 2010), Recency Weighting Technique (Joshi et al., 2007), Spam Filter Approach (Mizuno et al., 2007), Weighted Similarity Modelling (Nagwani and Verma, 2010), Association Rule Mining (Song et al., 2006)

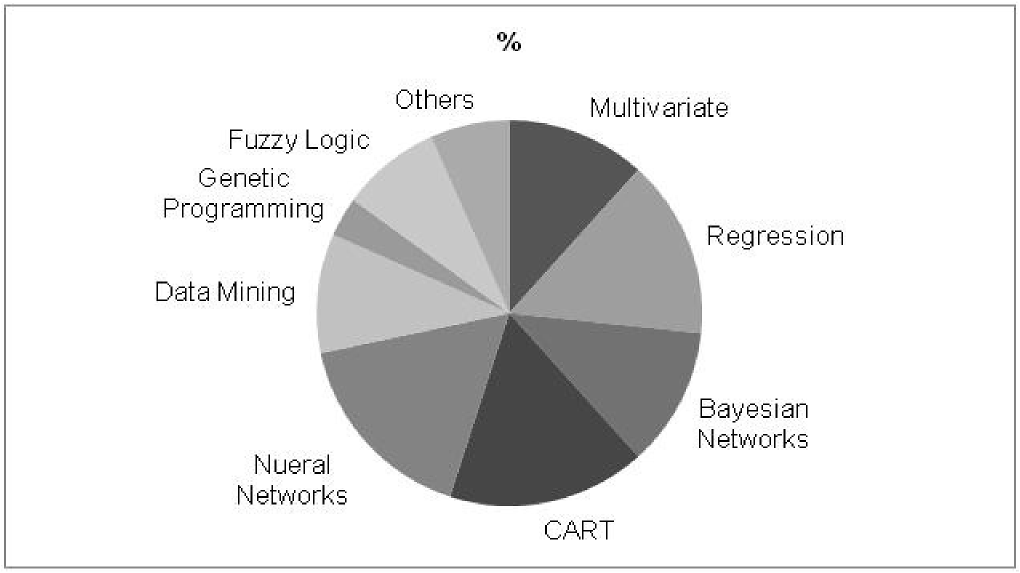

Figure 3. Breakup of different algorithms used to develop used in fault prediction models

Moreover software quality depends on people related factors like programmer skills, level of expertise, domain knowledge, etc (Antony and Fergusson, 2004). Hence the process performance models developed as per the guidelines of CMM frame work would be more useful for quantitatively managing the software development process.

Table 3. Management and evaluation practices of specific categories of software development

SL No.	Details
1	Comparison of quality of the software developed by distributed teams and collocated teams (Bird et al., 2009)
2	Development of metric based approach and software engineering metrics to ensure the quality of systems developed using commercial off the shelf components (COTS) (Sedigh-Ali et al., 2001; Rawashdeh and Matalkah, 2006)
3	Comparison of quality of software as a service (offering software using a subscription model) with software offered using perpetual licensing model (Choudhary, 2007)
4	Comparison of quality of open source applications with that of commercially developed software (Stamelos et al., 2002; Raymond, 2001; Aberdour, 2007)
5	Application of quality function deployment (QFD) on improving software quality (Liu, 2000; Islam and Hasin, 2014)
6	Application of fuzzy logic on quality improvement (Liu et al., 2006), effect of project management policies on software quality (Garcia et al., 2008), effect of various software process improvement (SPI) methodologies on software quality (Ashrafi, 2003)

Management and evaluation practices of specific categories of software development process

Apart from studying the effect or TQM, ISO, CMM and defect prediction modelling on software quality, lot of research has also been carried out in the field of management and evaluation practices of specific categories of software development process. The important among them are summarized in table 3.

CONCLUSION AND FUTURE RESEARCH WORK

The paper presented a literature review on software quality management practices. The review revealed that the important articles published recently can be classified into five groups namely those studying the impact of (i) TQM implementation on software quality, (ii) ISO 9000 certification on software quality, (iii) adoption of CMM framework on software quality, (iv) development of software quality prediction models and (v) Management and evaluation practices of specific categories of software development process.

The published articles suggested that there is positive correlation between TQM implementation and software quality as well as CMM frame work and software quality. These studies are carried out using empirical data collected through survey. The future researchers can fine tune these findings through correlating quantitative measures of software quality with that of TQM, ISO, CMM implementation. The researchers can also establish the correct relationship between ISO certification and software quality as the past studies showed that ISO certification has conflicting results.

A lot of research has been carried out on developing software quality prediction models to either predict the software faults or classify software modules as fault prone or not. Some of the drawbacks of these models are as follows: There is no one best model suitable for all types of software development. In fact the research on comparison of these models suggested that different models need to use for different type of scenarios. Moreover majority of these models use static code attributes like code complexity, etc as predictors. Many of these factors are not routinely measured by the project managers and also many of these factors may not be under the control of project managers. Most of these models used data available in the public domain like NASA website, etc. The research on developing fault prediction models using industry data are still in the nascent stage. The review showed that process performance models developed under the quantitative project management process area of CMM framework are more suitable for quantitatively manage the software development process. But these models are mostly based on regression or simulation techniques. The development of process performance models using different algorithms, especially the machine learning algorithms will be an important area of future research. Moreover the review showed that there is good correlation exit between software quality, productivity, cycle time, and development effort. Hence it is required to develop models or methodologies to quantitatively manage multiple performance characteristics simultaneously. Moreover many of the performance characteristics are qualitative. Hence the models for simultaneously monitoring of quantitative and qualitative performance characteristics will be another important area of future research.

REFERENCES

Aberdour M (2007). Achieving quality in open-source software. IEEE Software 24(1): 58-64.

Adam EE, Flores BE and Macias A (2001). Quality improvement practices and the effect on manufacturing firm performance: evidence from Mexico and the USA. Int. J. Prod. Res. 39(1): 46 – 63.

Afzal W and Torkar R (2008). A comparative evaluation of using genetic programming for predicting fault count data. In Third IEEE International Conference on Software Engineering Advances: 407-414.

Afzal W, Torkar R and Feldt R (2008). Prediction of fault count data using genetic programming. In IEEE International Multitopic Conference: 349-356.

Alkhafaji AF, Youssef MA and Sardessia R (1998). TQM, strategic management and business process re-engineering: the future challenge. International Journal of Technology Management 16(4-6): 383-392.

Al-Kilidar H, Cox K and Kitchenham B (2005). The use and usefulness of the ISO/IEC 9126 quality standard. In Proceedings of IEEE International Symposium on Empirical Software Engineering: 7-pp.

Amasaki S, Takagi Y, Mizuno O and Kikuno T (2003). A bayesian belief network for assessing the likelihood of fault content. In 14th IEEE International Symposium on Software Reliability Engineering: 215-226.

Antony J and Fergusson C (2004). Six Sigma in the software industry: results from a pilot study. Managerial Auditing Journal 19(8): 1025-1032.

Ashrafi N (2003). The impact of software process improvement on quality: in theory and practice. Information & Management 40(7): 677-690.

Balla K, Bemelmans T, Kusters R, and Trienekens J (2001). Quality through managed improvement and measurement (QMIM): Towards a phased development and implementation of a quality management system for a software company. Software Quality Journal, 9(3): 177-193.

Bezerra ME, Oliveira AL and Meira SR (2007). A constructive rbf neural network for estimating the probability of defects in software modules. In IEEE International Joint Conference on Neural Networks: 2869-2874

Bhattacharya TK, AlDiab-Zoubi T and Sukar A (1998). Application of total quality management concepts to a business school. International Journal of Technology Management 16(4-6): 520-531.

Bibi S, Tsoumakas G, Stamelos I and Vlahavas IP (2006). Software Defect Prediction Using Regression via Classification. In AICCSA: 330-336.

Binkley D, Feild H, Lawrie D and Pighin M (2007). Software fault prediction using language processing. Testing: IEEE Academic and Industrial Conference Practice and Research Techniques: 99-110.

Bird C, Nagappan N, Devanbu P, Gall H and Murphy B (2009). Does distributed development affect software quality?: an empirical case study of Windows Vista. Communications of the ACM 52(8): 85-93.

Camuffo M, Maiocchi M and Morselli M (1990). Automatic software test generation. Information and Software Technology 32(5): 337-346.

Carroll J (1995). The application of total quality management to software development. Information Technology & People 8(4): 35-47.

Ceylan E, Kutlubay FO and Bener AB (2006). Software defect identification using machine learning techniques. In 32nd IEEE EUROMICRO Conference on Software Engineering and Advanced Applications: 240-247.

Challagulla VU, Bastani FB and Yen IL (2006).'A unified framework for defect data analysis using the mbr technique. In 18th IEEE International Conference on Tools with Artificial Intelligence: 39-46.

Choudhary V (2007). Comparison of software quality under perpetual licensing and software as a service. Journal of Management Information Systems 24(2):141-165.

Coallier F (1994). How ISO 9001 fits into the software world. IEEE Software 11(1): 98-100.

Cortada JW (1995). TQM for information systems management: quality practices for continuous improvement, McGraw-Hill, Inc.

Cruz AC and Ochimizu K (2009). Towards logistic regression models for predicting fault-prone code across software projects. In 3rd IEEE International Symposium on Empirical Software Engineering and Measurement: 460-463.

De Carvalho AB, Pozo A and Vergilio SR (2010). A symbolic fault-prediction model based on multiobjective particle swarm optimization. Journal of Systems and Software 83(5): 868-882.

De Carvalho AB, Pozo A, Vergilio S and Lenz A (2008). Predicting fault proneness of classes trough a multiobjective particle swarm optimization algorithm. In 20th IEEE International Conference on Tools with Artificial Intelligence 2: 387-394.

El Emam K, Benlarbi S, Goel N and Rai SN (2001). Comparing case-based reasoning classifiers for predicting high risk software components. Journal of Systems and Software 55(3): 301-320.

Elish KO and Elish MO (2008). Predicting defect-prone software modules using support vector machines. Journal of Systems and Software 81(5): 649-660.

Fenton N and Bieman J (2014). Software metrics: a rigorous and practical approach, CRC Press.

Fenton N, Neil M, Marsh W, Hearty P, Marquez D, Krause P and Mishra R (2007). Predicting software defects in varying development lifecycles using Bayesian nets. Information and Software Technology 49(1): 32-43.

Fenton NE and Neil M (1999). A critique of software defect prediction models. IEEE Transactions on Software Engineering 25(5): 675-689.

Franch X and Carvallo JP (2003). Using quality models in software package selection. IEEE Software 20(1): 34-41.

Ganesan K, Khoshgoftaar TM and Allen EB (2000). Case-based software quality prediction. International Journal of Software Engineering and Knowledge Engineering 10(2): 139-152.

García MNM, Román IR, Peñalvo FJG and Bonilla MT (2008). An association rule mining method for estimating the impact of project management policies on software quality, development time and effort. Expert Systems with Applications 34(1): 522-529.

Gong B, Yen DC and Chou DC (1998). A manager's guide to total quality software design. Industrial Management & Data Systems 98(3): 100-107.

Guo L, Cukic B and Singh H (2003). Predicting fault prone modules by the dempster-shafer belief networks. In Proceedings of 18th IEEE International Conference on Automated Software Engineering: 249-252.

Guo L, Ma Y, Cukic B and Singh H (2004). Robust prediction of fault-proneness by random forests. In 15th IEEE International Symposium on Software Reliability Engineering: 417-428.

Hansen B, Rose J, and TjøRnehø JG (2004). Prescription, description, reflection: the shape of the software process improvement field. International Journal of Information Management 24(6): 457-472.

Hao Y and Zhang YF (2011). Statistical prediction modeling for software development process performance. In 3rd IEEE International Conference on Communication Software and Networks (ICCSN): 703-706.

Harter DE, Krishnan MS and Slaughter SA (2000). Effects of process maturity on quality, cycle time, and effort in software product development. Management Science 46(4): 451- 466.

Herbsleb J, Zubrow D, Goldenson D, Hayes W and Paulk M (1997). Software quality and the capability maturity model. Communications of the ACM 40(6): 30-40.

Hribar L and Duka D (2010). Software component quality prediction using KNN and Fuzzy logic. In Proceedings of the 33rd IEEE International Convention MIPRO: 402-408.

IDC (2003). Worldwide black book Q2.

Islam KD and Hasin AA (2014). A quality transfer of the requirements of teachers into technical requirements: Use of House of Quality (HOQ) matrix in Quality Function Deployment (QFD). Merit Research Journal of Business and Management 2(1): 7 – 12.

ISO/IEC International Standard 9126 (1991). Information Technology - Software product Evaluation - Quality Characteristics and Guidelines for their use, International Standards Organization.

Issac G, Rajendran C and Anantharaman RN (2004). A conceptual framework for total quality management in software organizations. Total Quality Management & Business Excellence 15(3): 307-344.

Jenner MG (1995). Software quality management and ISO 9001: how to make them work for you, John Wiley & Sons, Inc.

Jin C, Dong EM and Qin LN (2010). Software fault prediction model based on adaptive dynamical and median particle swarm optimization. In 2nd IEEE International Conference on Multimedia and Information Technology 1: 44-47.

John B (2012). Modeling the Defect Density of Embedded System Software Using Bayesian Belief Networks: A Case Study. Software Quality Professional 14(3): 39 – 45.

John B and Kadadevaramath RS (2014). A methodology for achieving the design review defect density goals in software development process. International Journal of Manufacturing, Industrial & Management Engineering 2(1): 181 – 191.

Joshi H, Zhang C, Ramaswamy S and Bayrak C (2007). Local and global recency weighting approach to bug prediction. In Proceedings of the Fourth IEEE International Workshop on Mining Software Repositories: 33.

Jung HW, Kim SG and Chung CS (2004). Measuring software product quality: A survey of ISO/IEC 9126. IEEE software 5: 88-92.

Kanji GK and Asher M (1999). 100 Methods for Total Quality Management, Sage Publishers, New Delhi. India.

Kanmani S, Uthariaraj VR, Sankaranarayanan V and Thambidurai P (2004). Object oriented software quality prediction using general regression neural networks. ACM SIGSOFT Software Engineering Notes 29(5): 1-6.

Kanmani S, Uthariaraj VR, Sankaranarayanan V and Thambidurai P (2007). Object-oriented software fault prediction using neural networks. Information and software technology 49(5): 483-492.

Kaur A and Malhotra R (2008). Application of random forest in predicting fault-prone classes. In IEEE International Conference on Advanced Computer Theory and Engineering: 37-43.

Kemerer CF (1997). Software project management readings and cases, McGraw-Hill, New York, USA.

Khoshgoftaar TM and Allen EB (1999). Logistic regression modeling of software quality. International Journal of Reliability, Quality and Safety Engineering 6(4): 303-317.

Khoshgoftaar TM and Munson JC (1990). Predicting software development errors using software complexity metrics. IEEE Journal on Selected Areas in Communications 8(2): 253-261.

Khoshgoftaar TM and Seliya N (2002). Improving usefulness of software quality classification models based on boolean discriminant functions. In Proceedings of 13th IEEE International Symposium on Software Reliability Engineering: 221-230.

Khoshgoftaar TM and Seliya N (2002). Tree-based software quality estimation models for fault prediction. In Proceedings of Eighth IEEE Symposium on Software Metrics: 203-214.

Khoshgoftaar TM and Seliya N (2003a). Analogy-based practical classification rules for software quality estimation. Empirical Software Engineering 8(4): 325-350.

Khoshgoftaar TM and Seliya N (2003b). Software quality classification modeling using the SPRINT decision tree algorithm. International Journal on Artificial Intelligence Tools 12(3): 207-225.

Khoshgoftaar TM, Allen EB, Hudepohl JP and Aud SJ (1997). Application of neural networks to software quality modeling of a very large telecommunications system. IEEE Transactions on Neural Networks 8(4): 902-909.

Khoshgoftaar TM, Allen EB, Jones WD and Hudepohl JP (1999). Classification tree models of software quality over multiple releases. In Proceedings of 10th IEEE International Symposium on Software Reliability Engineering: 116-125.

Khoshgoftaar TM, Gao K and Szabo RM (2001). An application of zero-inflated Poisson regression for software fault prediction. In Proceedings of 12th IEEE International Symposium on Software Reliability Engineering: 66-73.

Khoshgoftaar TM, Pandya AS and Lanning DL (1995). Application of neural networks for predicting program faults. Annals of Software Engineering 1(1): 141-154.

Khoshgoftaar TM, Seliya N and Sundaresh N (2006). An empirical study of predicting software faults with case-based reasoning. Software Quality Journal 14(2): 85-111.

Knab P, Pinzger M and Bernstein A (2006). Predicting defect densities in source code files with decision tree learners. In Proceedings of the 2006 international workshop on Mining software repositories: 119-125.

Koru A and Liu H (2005). Building effective defect-prediction models in practice. IEEE Software 22(6): 23-29.

Krishnan MS (1997). Cost and quality considerations in software product management, Carnegie Mellon University.

Li EY, Chen HG and Cheung W (2000). Total quality management in software development process. The Journal of Quality Assurance Institute 4(1): 35 - 41.

Liu XF (2000). Software quality function deployment. IEEE Potentials 19(5): 14-16.

Liu XF, Kane G and Bambroo M (2006). An intelligent early warning system for software quality improvement and project management. Journal of Systems and Software. 79(11): 1552-1564.

Mahaweerawat A, Sophatsathit P and Lursinsap C (2007). Adaptive self-organizing map clustering for software fault prediction. In Fourth International Joint Conference on Computer Science and Software Engineering: 35-41.

Mair C, Kadoda G, Lefley M, Phalp K, Schofield C, Shepperd M and Webster S (2000). An investigation of machine learning based prediction systems. J. Systems and Software 53(1): 23-29.

Menzies T, Di Stefano JS and Chapman M (2003). Learning early lifecycle IV & V quality indicators. In Proceedings of Ninth IEEE International Software Metrics Symposium: 88 - 96.

Menzies T, Greenwald J and Frank A (2007). Data mining static code attributes to learn defect predictors. IEEE Transactions on Software Engineering 33(1): 2-13.

Mizuno O, Ikami S, Nakaichi S and Kikuno T (2007). Spam filter based approach for finding fault-prone software modules. In Proceedings of the Fourth International Workshop on Mining Software Repositories: 4.

Mooney JG, Gurbaxani V and Kraemer KL (1996). A process oriented framework for assessing the business value of information technology. ACM SIGMIS Database 27(2): 68-81.

Morasca S and Ruhe G (1996). A comparative study of two techniques for analyzing software measurement data. In Proceedings of Annual Meeting, ISERN.

Munson JC and Khoshgoftaar TM (1990). Regression modelling of software quality: empirical investigation. Information and Software Technology 32(2):106-114.

Munson JC and Khoshgoftaar TM (1992). The detection of fault-prone programs', IEEE Transactions on Software Engineering 18(5): 423-433.

Nagappan N, Williams L, Osborne J, Vouk M and Abrahamsson P (2005). Providing test quality feedback using static source code and automatic test suite metrics. In 16th IEEE International Symposium on Software Reliability Engineering: 10-pp.

Nagwani NK and Verma S (2010). Predictive data mining model for software bug estimation using average weighted similarity. In IEEE 2nd International Advance Computing Conference: 373-378.

Neil M (1992). Multivariate assessment of software products. J. Software Testing, Verification and Reliability 1(4): 17-37.

Padberg F, Ragg T and Schoknecht R (2004). Using machine learning for estimating the defect content after an inspection. IEEE Transactions on Software Engineering 30(1): 17-28.

Pai GJ and Dugan JB (2007). Empirical analysis of software fault content and fault proneness using Bayesian methods. IEEE Transactions on Software Engineering 33(10): 675-686.

Parzinger MJ and Nath R (2000). A study of the relationships between total quality management implementation factors and software quality. Total Quality Management 11(3): 353-371.

Paulk MC (1993). Comparing ISO 9001 and the capability maturity model for software. Software Quality J. 2(4): 245-256.

Pearson MJ, McCahon CS and Hihgtower RT (1995). Total quality management: are information systems managers ready? Information and Management 29(5): 251 – 263.

Phan DD, George JF and Vogel DR (1995). Managing software quality in a very large development project. Information & management 29(5): 277-283.

Porter AA and Selby RW (1990) Evaluating techniques for generating metric-based classification trees. Journal of Systems and Software 12(3): 209-218.

Pressman RS (2005). Software engineering: a practitioner's approach, Palgrave Macmillan.

Rawashdeh A and Matalkah B (2006). A new software quality model for evaluating COTS components. J. Comp. Sci. 2(4): 373-381.

Raymond ES (2001). The Cathedral & the Bazaar: Musings on linux and open source by an accidental revolutionary, O'Reilly Media, Inc.

Reformat M (2003). A fuzzy-based meta-model for reasoning about the number of software defects. In Fuzzy Sets and Systems—IFSA

2003: 644-651.

Reformat M, Pedrycz W And Pizzi NJ (2003). Software quality analysis with the use of computational intelligence. Information and Software Technology 45(7): 405-417.

Samson D And Terziovski M (1999). The relationship between Total Quality Management practices and operational performance. Journal of Operations Management 17(4): 393 – 409.

Sandhu PS, Kaur M and Kaur A (2010). A Density Based Clustering approach for early detection of fault prone modules. In 2010 IEEE International Conference on Electronics and Information Engineering 2: V2-525.

Schneidewind N F (2001). Investigation of logistic regression as a discriminant of software quality. In Proceedings of IEEE Seventh International Software Metrics Symposium. 328-337.

Sedigh-Ali S, Ghafoor A and Paul R (2001). Metrics-guided quality management for component-based software systems. In 25th IEEE Annual International Computer Software and Applications Conference: 303-308.

Shem S, Eddie L and Kellys S (2015). Assessing the role of standards in enhancing the competitiveness of locally manufactured products in Zambia. Merit Research J. Bus. Manag. 3(2): 16 – 28.

Sherriff M, Heckman SS, Lake M and Williams L (2007). Identifying fault-prone files using static analysis alerts through singular value decomposition. In Proceedings of the 2007 conference of the center for advanced studies on Collaborative research: 276-279..

Shiffler G (2003). Gartner Dataquest Market Databook March 2003 Update, Gartner.

Sibisi M and Van Waveren CC (2007). A process framework for customising software quality models. In Proceedings of the IEEE AFRICON: 1-8.

Singh Y, Kaur A and Malhotra R (2008). Predicting software fault proneness model using neural network. In Product-Focused Software Process Improvement: 204-214. Springer Berlin Heidelberg.

Song Q, Shepperd M, Cartwright M and Mair C (2006). Software defect association mining and defect correction effort prediction. IEEE Transactions on Software Engineering 32(2): 69-82.

Stamelos I, Angelis L, Oikonomou A and Bleris GL (2002). Code quality analysis in open source software development. Information Systems Journal 12(1): 43-60.

Subramanian GH, Jiang JJ and Klein G (2007). Software quality and IS project performance improvements from software development process maturity and IS implementation strategies. J. Systems and Software 80(4): 616-627.

Succi G, Stefanovic M and Pedrycz W (2001). Advanced statistical models for software data, Department of Electrical and Computer Engineering, University of Alberta, Canada, http://www. unibz. it/web4archiv/objects/pdf/cs_library/2/AdvancedStatisticalModelsfor Softwaredata. pdf.

Suryn W, Abran A, and April A (2003). ISO/IEC SQuaRE the second generation of standards for software product quality.

Tamura S (2009). Integrating CMMI and TSP/PSP: Using TSP Data to Create Process Performance Models (No. CMU/SEI-2009-TN-033), Carnegie-Mellon University, Software Engineering Institute, Pittsburgh.

Tao W and Wei-hua L (2010). Naive bayes software defect prediction model. In IEEE International Conference on Computational Intelligence and Software Engineering: 1-4.

Team CP (2006). CMMI for Development, version 1.2.

Thwin MMT and Quah TS (2005). Application of neural networks for software quality prediction using object-oriented metrics. Journal of systems and software 76(2): 147-156.

Turhan B and Bener A (2007). A multivariate analysis of static code attributes for defect prediction. In Seventh IEEE International Conference on Quality Software: 231-237.

US General Accounting Office (1979). Contracting for computer software development, FGMSD-80.4, Washington DC, USA.

Walrad C and Moss E (1993). Measurement: the key to application development quality. IBM Systems Journal 32(3): 445-460.

Weber CV, Curtis B and Chrissis MB (1994). The capability maturity model: Guidelines for improving the software process, Addison-Wesley, Reading, MA.

Xu Z, Khoshgoftaar TM and Allen EB (2000). Prediction of software faults using fuzzy nonlinear regression modeling. In Fifth IEEE International Symposim on High Assurance Systems Engineering: 281-290.

Yang B, Yao L and Huang HZ (2007). Early software quality prediction based on a fuzzy neural network model. In Third International Conference on Natural Computation 1: 760-764.

Yang HY (2001). Software quality management and ISO 9000 implementation. Industrial Management & Data Systems 101(7):329-338.

Yang W and Li L (2008). A rough set model for software defect prediction. In IEEE International Conference on Intelligent Computation Technology and Automation 1: 747-751.

Yuan X, Khoshgoftaar TM, Allen EB and Ganesan K (2000). An application of fuzzy clustering to software quality prediction. In Proceedings of 3rd IEEE Symposium on Application-Specific Systems and Software Engineering Technology: 85-90.

Zardony MA and Tumanic RE (1992). Zero-defects software: the total quality management approach to software engineering. Chief Information Officer J. 4(4): 10-16.

Zhu D and Wu Z (2009). The Application of Gray-Prediction Theory in the Software Defects Management. In IEEE International Conference on Computational Intelligence and Software Engineering: 1-5.

Zimmerman LV (2001). Plan for training to ensure software quality. AACE International Transactions: IT51.