Software Metrics: Investigating Success Factors, Challenges, Solutions And New Research Directions

Hina Noor, Dr. Babur Hayat, AbuBakar Hamid, Tamoor Wakeel, Rashida Nasim

Abstract: Software metrics is becoming important every day. A measure of software characteristics which are countable or measurable is known as software metrics. Software metrics are very important because they are used for many purposes for example, estimating efficiency, arranging work things and estimating execution of software etc. Software product and software development process specific attributes are measured by software metrics. Many metrics related to coupling, cohesion etc. have been defined. This paper investigates success factors and challenges in the implementation of software metrics and also tries to investigate solutions to those challenges. The paper explains new research directions to software metrics as well which really a need of today.

Index Terms: Software metrics, Measurements, Types of metrics, Success factors of software metrics, Challenges in implementation of software metrics, New research directions in software metrics.

1. INTRODUCTION

IN software, measurements are done using software metrics. A software metric is a measurement measure of the degree to which a software program or procedure has certain capital. Even if a metric is not a measure (metrics are functions, while measurements are the numbers obtained through metric application) [1]. Accurate measurement for all engineering fields is a requirement and software engineering is no different. For decades engineers and academics are persuading to communicate numerical software functionality to enable quality assurance of software. A broad variety of quality software measures have been created and there are various methods to obtain measures from system representations [2], [20]. There are several metrics inside the software development cycle and are all connected to each other. The software metrics are linked to the four management functions: planning, organizing, controlling or improving. Software metrics are perfect for the teams of management because they provide a quick way to monitor, set targets and evaluate success [1].

- *Hina Noor is currently pursuing MS- Software Engineering at University of Lahore, Gujrat campus, Pakistan, E-mail: hinanoorm006@gmail.com*
- *Dr. Babur Hayat Malik Assistant Professor of CS & IT department at University of Lahore, Gujrat campus, Pakistan, E-mail: baber.hayat@cs.uol.edu.pk*
- *AbuBakar Hamid is currently pursuing MS-SE at University of Lahore ,Gujrat Campus, Pakistan, Email: abubakarhamid44@gmail.com*
- *Tamoor Wakeel has done MS-IT at University of Lahore ,Gujrat Campus, Pakistan, Email: Tamoor.wakeel72@gmail.com*
- *Rashida Nasim is currently pursuing MS-SE at University of Lahore, Gujrat Campus, Pakistan, Email: Rashitarar94@gmail.com*

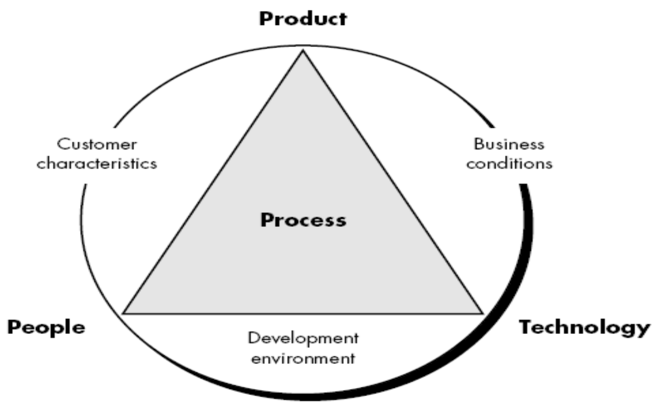


Fig1.

Software metrics is a logical term for applying quantitative metric values to the instances of a software program. We also believe that software metric tools enforce software metrics as described by them [3]. The usage of software metrics is specifically meant to assess the product output. We are, though, often used not only to forecast product or process output but also to enhance efficiency. Major uses of software metrics are Project estimation and progress checking, Evaluation of work products, Improvement of processes by condition monitoring, and experimental validation of best practice [2], [21].

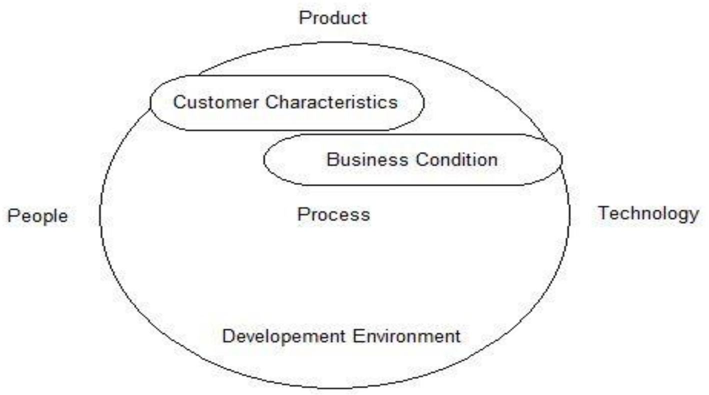


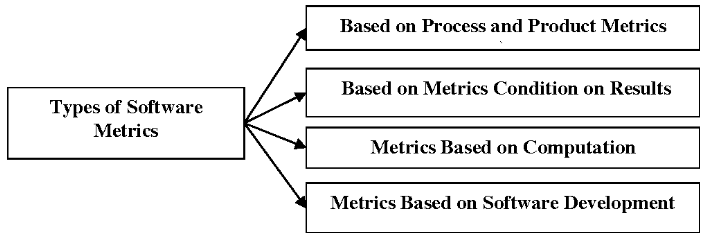
Fig2.

The intend of this paper is to investigate the success factors, failures/ challenges in the implementation of software metrics. We also try to suggest the solutions to overcome these challenges. We also try to derive new research directions to software metrics for the betterment in software engineering. The paper is split down into five major sections. Section 2, describes types of metrics. Section 3 deals with the literature review. The section 4 summarizes success factors in software metrics. The 5 section explains the challenges in the implementation of software metrics and also suggest their solutions. In Section 6 we describe new direction in the research of software metrics. Section 7 concludes the whole discussion.

2 TYPES OF SOFTWARE METRICS

Software metrics based on different categories [1]:

- Process and product metrics
- Metrics condition on results
- Computation
- Software development



Types of Software Metrics
Based on Process and Product Metrics
Based on Metrics Condition on Results
Metrics Based on Computation
Metrics Based on Software Development

Fig3.

2.1 Process metrics

Process metrics evaluate the viability and nature of programming process; decide development of the procedure, exertion required all the while, and adequacy of imperfection expulsion during advancement [1].

2.2 Product metrics

Product metrics are the estimation of work item created during various periods of software development [1].

Process Metrics
Product Metrics
Project Metrics

Fig4.

2.3 Project metrics

Project metrics portray venture qualities and execution. Models incorporate the number of software engineers, the staffing design over the existing pattern of the cost, timetable, and efficiency. Project metrics illustrate the project characteristics and their execution [1], [18].

3. LITERATURE REVIEW

Reena Sharma, R.S.Chhillar discussed some flaws in software metrics and proposed metrics uses the concept of algorithm. According to them it is challenging to pick the right metrics for particular software and, second, most metrics only follow the requirements process. The study explores the different Techniques, their benefits and demerits and efforts to suggest a modern approach for evaluating the quality of the implementation cycle. The suggested metrics using the Genetic Algorithms concept which is based on the theory of natural selection. Thus, the research intends to incorporate natural selection methods to assess software efficiency [1]. Rüdiger Lincke, Jonas Lundberg and Welf Löwe reveal in their paper that the current software metric methods differently view and enforce the meanings of object-oriented software metrics. With three software systems with various sizes they measured metrics values using the same collection with basic metrics. Measurements indicate All the metrics are tool-dependent on the same working system and calculations. We have set up a (simple) app quality model based on the chosen metrics for the "maintainability" [2]. Jernej Novak discussed the issue that whether different software metrics tools give us different results. This article is targeted at to test software metric measurements and decide whether the tools are producing the same performance. We'd like to find out whether using a particular method will produce specific outcomes and therefore deliver new decisions [3]. Norman E Fenton and Martin Neil used approach of Bayesian Belief nets, which are progressively observed as the best methods for taking care of dynamic under vulnerability. They firstly described the successes about software metrics and its correlated actions, secondly described failures about software metrics and then conclude new directions. They also concluded that in reality, the BBN technique is used on actual projects and earns highly favorable feedback. We assume metrics study is a significant path forward [4]. Norman Fenton concluded some new direction of software metrics. He proposed some successes about software metrics that are: The reported empirical findings and models contributed to some very useful criteria for measuring, Especially of IT departments have any sort of metrics program, even though they do not specifically regard them as such and where administrators have more influence of projects as they are more aware of what is happening, Metrics was a phenomenal achievement measured by the research / academic performance. There are thousands of papers written, scores of textbooks and many conferences devoted to them. Computer metrics are now recognized as central information of software engineering [5]. Carol A. Dekkers and Patricia A. McQuaid proposed basic problems of software metrics. They proposed that the unsuccessful measurement programs, owing to a mixture of three specific issues, frequently fail are: A fundamental misconception of the hidden hypothesis of estimations and what is estimated, the confusion of estimation information prompts unintended reactions that are devastating the advancement and achievement of the association, and Disregard for human factors (how the societal transition in metrics impacts people) and app developers' awareness level [6]. Robert B. Grady and Hewlett-Packard worked on successful implementation of software metrics. They concluded major uses of software metrics in their paper very clearly [8]. Tracy Hall and Norman Fenton identified Consensus on metric system criteria in their research , i.e. gradual introduction, consistency, quality of measurements, automatic data collection, dedicated metrics team and goal-oriented approach [7].

3.1 Research Questions

1. What are the success factors and challenges in the implementation of software metrics?
2. What are the solutions to the challenges that we faced in the implementation of software metrics?
3. What are the new research directions in the software metrics?

4 SUCCESS FACTORS OF SOFTWARE METRICS

The idea of software metrics observing and survey is to evaluate the nature of the present item or activity, increment the quality and gauge the yield until the product advancement venture is finished. Metrics are an important component in quality control, monitoring, testing, efficiency, and cost assessment and are essential to all developers and members of production teams [9].

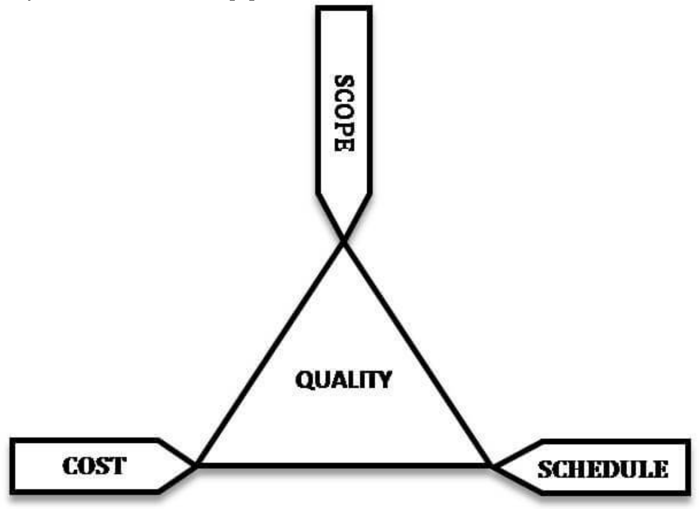


Fig5.

The volume of software metrics interference currently taking place in business has gone through the roof. Many big IT companies, and even other smaller ones, have 'system' software metrics in operation, as compared to last few years [4]. Significant uses of software metrics are growing value for engineers and project managers and improved usefulness for task groups and upper management [8].

4.1 Project estimation and progress monitoring

Today there are really hundreds of software estimation tools accessible to project managers. Such tools compensate for several potential variables on the project [2], [15]. Already most of us are experienced at calculating the correct values for these variables even as we speak about complete project schedules. Project estimation monitoring and progress also make us capable of understand the basis for estimates, the bottom line, successful usage, and monitoring progress against estimates for projects very clearly [8]. In short it is undoubted a huge success of software metrics because it gives much success to our software projects.

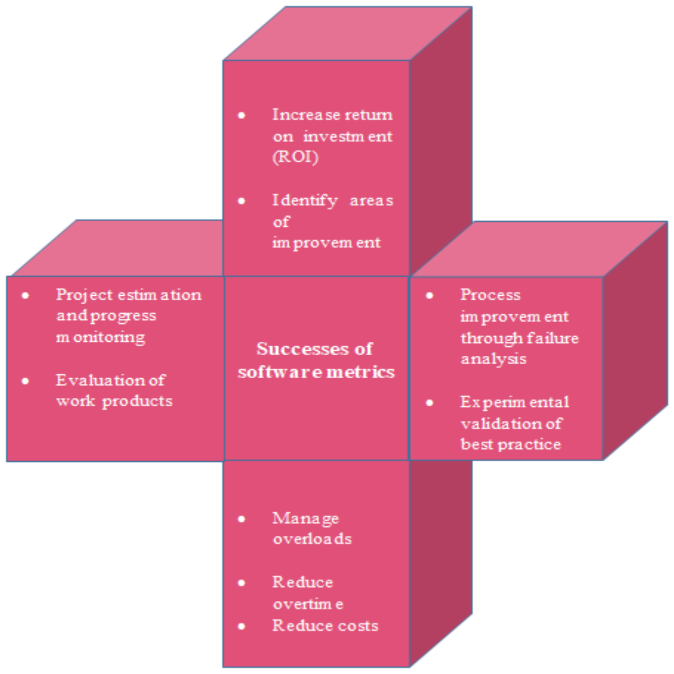

Increase return on investment (ROI)
Identify areas of improvement
Project estimation and progress monitoring
Evaluation of work products
Successes of software metrics
Process improvement through failure analysis
Experimental validation of best practice
Manage overloads
Reduce overtime
Reduce costs

Fig6. *Success factors of software metrics.*

4.2 Evaluation of work products

A work product is a middle or last yield which characterizes the plan, the procedure, creation or testing of any segment of a deliverable or marketable item. The thing to note is the concept of removing non-valuable measurements practically from every work product must be remember. Successfully software metrics allow us to reduce cyclomatic complexity and design complexity and allow us to give more attention on successful usage. [8]

4.3 Process improvement through failure analysis

Failure analysis, Identifying and removing large causes of defects offer the greatest short-term opportunities for progress guidelines. Software metrics allows us to improve our software process through failure analysis so that it may not cause of a bigger failure, in this way we can improve our development process as well. There are two types of failure analysis which software metrics provide us project defect pattern and software process defect pattern [8].

4.4 Experimental validation of best practice

This software metric utilization of information was the most effective of the above three mentioned. People also validated the effectiveness of major development activities (e.g. “prototyping”, "the coupling, the cohesion," restricting complexity, inspections and testing methods, and models of reliability). This recognition would result in a more rapid and universal adoption of these "correct" approaches [8]. Certainly such metrics gave the most advantages to project managers. The primary model here is the thing that elevated level supervisors like to see. Studies demonstrated that the general execution of code perusing/code assessments has been 4.4 occasions higher than other yield testing strategies. This evidence lets project leaders schedule inspections for their programs and reassure their engineers by demonstrating results from inspections [8].

4.5 Increase return on investment (ROI)

The Return on Investment (ROI) metric is a typical device to quantify the budgetary effects of speculations and exercises. Rate of profitability (ROI) is an important measurement used to decide a venture's replacement to assess the efficiency of an assortment of explicit speculations. To quantify the ROI, a speculation's benefit (or return) is part by the venture costs. The outcome is communicated in extent or proportion The ROI determined is a proportion or rate which thinks about net increases to net expenses. Return on investment (ROI) metrics measures profitability step-by-step compared to other methods [9], [16]. Software metrics are an essential component in assurance of quality; monitoring, analysis, efficiency, and price analysis so in this way metrics increase ROI. Metrics are valuable to all developers and members of development teams.

4.6 Identify areas of improvement

Software metrics include an overview of the effect of the product creation programs choices taken. This lets managers determine priorities and success targets, and prioritize them. In this way they can pay attention on identifying areas of improvement [8], [9].

4.7 Manage workloads

For improved group viability, supervisors can utilize programming measurements to characterize, compose, screen and impart any issues. It makes for efficient communication and makes for task identification and prioritization inside app development programs and helps to manage workload between development team [4], [9].

4.8 Reduce overtime

Programming improvement groups can utilize programming measurements to impart programming advancement tasks progress, detect and fix problems, and track, enhance, and maintain their workflow better. In this way it reduces the overtime [7], [9].

4.9 Reduce costs

By using software metrics the managers can spot issues with the program earlier, the faster and less costly the cycle of troubleshooting. In this ways metrics reduce cost of the software development process [4], [9].

5 CHALLENGES, AND SUGGESTIONS TO OVERCOME THESE CHALLENGES

Efficient measurement programs, due to a variation of three simple issues, sometimes fail [6]:

- A profound misconception of the basic measurement principle and what steps are being taken.
- The misconception of estimation information adds to unexpected symptoms that are undermining the growth and success of the organization.

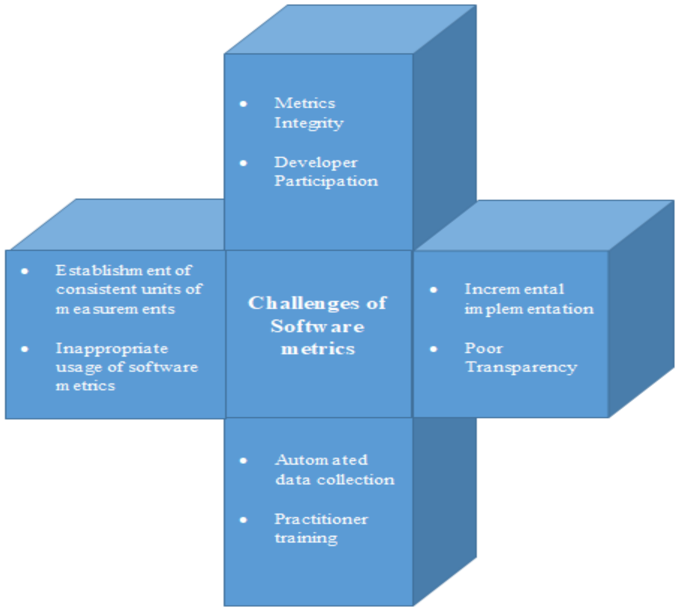

Metrics Integrity
Developer Participation
Establishment of consistent units of measurements
Inappropriate usage of software metrics
Challenges of Software metrics
Incremental implementation
Poor Transparency
Automated data collection
Practitioner training

Fig7. *Challenges of software metrics*

- Disregard for human factors (how the societal shift in evaluation would impact people) and the information strength of software growth. In other terms, administrators sometimes interpret measures from a solely technological viewpoint and may not represent the extent of human engagement or information strength (mental focus, subject matter experience, and high level of communication).First examination of the steps and their future, functional effect, before gathering and acting on them is important. Common sense is the most essential instrument to apply. As developer Cem Kaner states: “The invisibility of underlying measurement models has led people to use inadequate and inappropriate 'metrics,' deluding themselves and causing havoc to their employees” [10].

Most work into academic metrics is fundamentally unrelated to industry needs. Irrelevance here is two-tiered: irrelevance in scope and irrelevance in content [4].

5.1.1 Establishment of consistent units of measurement

It is important to develop a measuring system and specific units of measurement that are to be utilized during the project existence. But establishing it sometimes becomes challenging because depending on which calculating system is used a single program package may have two somewhat different LOC counts. This makes it hard to contrast software and code lines or different measurements without a standard definition [4], [9].

5.1.2 Inappropriate usage of software metrics

There is likewise an issue about whether to decipher the product measurements. In the event that an association utilizes profitability measurements that show the measure of code and blunders, software engineers may oppose tending to troublesome issues to keep up their LOC and check down mistakes. Software engineers who compose a gigantic measure of essential code will have astounding quantities of benefit however not extraordinary abilities in structuring programming [9], [17].

5.1.3 Incremental implementation
Implementation over time of a metrics program bears considerable risk [7].

5.1.4 Poor Transparency
For practitioners, the metrics program has to be obvious. Practitioners need to comprehend what information is gathered, why it is gathered, and how it is utilized, but they lake this information so that it causes poor transparency [7].

5.1.5 Metrics integrity
Practitioners should trust in the data they collect. They have to believe sampling, sampling properly and not getting "fiddled" is smart [7].

5.1.6 Developer participation
Developers will be interested in designing of the metrics program. Buy-in is more likely with a high level of developer participation, as with implementing a more incisive metrics program. But developer did not focus on it properly [7].

5.1.7 Automated data collection
Any place conceivable that ought to be finished. Limiting additional work for designers regularly limits metric obstruction by engineers. It likewise guarantees the gathered information is bound to be accurate. Practitioners still face issues in it [7], [17].

5.1.8 Practitioner training
Appropriate training is not giving in different levels in organizations and there is no awareness rising to training in statistical analysis techniques to practitioners. Studies suggest that a measurement system with a number of qualified practitioners is more likely to achieve success [7].

5.2 FACTORS TO CONSIDER IN CHOOSING A MEASURE TO OVERCOME CHALLENGES

We find seven factors which can lead to highly successful software measurement programs. We can also call them characteristics for successful measurement programs [6], [11].

I

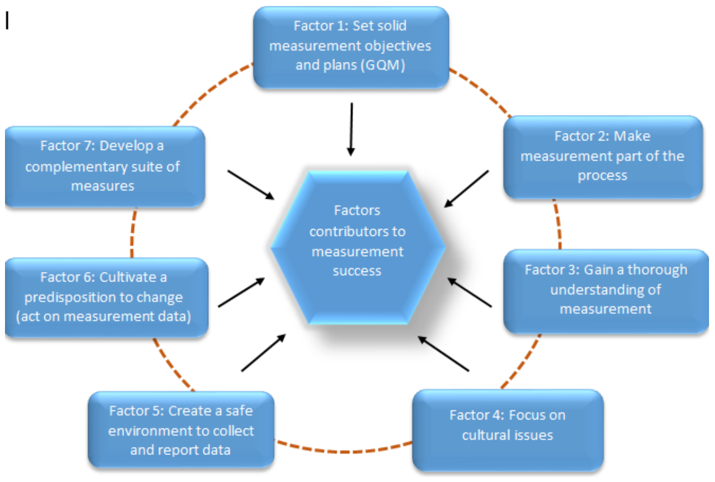

Factor 1: Set solid measurement objectives and plans (GQM)
Factor 2: Make measurement part of the process
Factor 3: Gain a thorough understanding of measurement
Factor 4: Focus on cultural issues
Factor 5: Create a safe environment to collect and report data
Factor 6: Cultivate a predisposition to change (act on measurement data)
Factor 7: Develop a complementary suite of measures
Factors contributors to measurement success

Fig8. *contributors to measurement success factors*

These are factors that are must to consider in choosing software measurements [6], [11].
Now Table 1 is going to give concepts of these factors by providing some description.

TABLE1

FACTORS TO HIGHLY SUCCESSFUL MEASUREMENT PROGRAMS

Factors	Description
Set solid measurement objectives and plans	Ensure the performance of the measurement system as a full implementation initiative, including comprehensive project management.
Make measurement part of the process	Point your objectives, questions and measurements to help basic choice taking procedures and upgrade the utilization of existing information assortment practices to satisfy your requirements.
Gain a thorough understanding of measurement	Managers may be nervous regarding calculation outcomes; they must understand that measurement requires a cultural change — in reality, rather than an intestinal move to management.
Focus on cultural issue	People are the most crucial component in the performance of software measurement, organizational calculation strategies should not skip over cultural concerns.
Create a secure framework for accurate data collection and reporting	Provided with a clear description of the phase of growth, IT managers will create disciplinary steps that can contribute to actual phase change.
Grow a predisposition to change	Associations with good system estimates react effectively to what the knowledge estimates educates them.
Develop an accompanying suite of measures	Software estimation program's arranging and plan, it is beneficial to choose a couple of integral measurements to screen and report the consequences of procedure improvement.

5.3 (GQM) Model
Victor Basili and David Weiss developed the Goal-Question-Metric (GQM) approach which is the best model for addressing many of these issues [6], [12]. Basili's GQM method stated that firstly sets the goals (purpose and scope) to be achieved through measurement, and then Define the quantifiable evidence is needed to decide whether progress is being made towards the goals [13]. Questions provide a guide by identifying the appropriate qualities and characteristics for calculation and metrics. Specific questions can discuss and mitigate possible side-effects of the measures collected [14], and then describes the individual measurement data properties, meanings and frequency of occurrence (collection). At this point you decide the principle of measurement, mathematical architecture and the applicability of measurement results [6], [12]

6 NEW RESEARCH DIRECTIONS IN SOFTWARE METRICS

- What we really need to achieve is an approach that incorporates our true objective for Software metrics: genuine cause and effect relationship, multiple types of evidence, expert judgment. There should be models in software metrics as well which deals with the following terms (as shown in figure 9) during the development of software project [5].

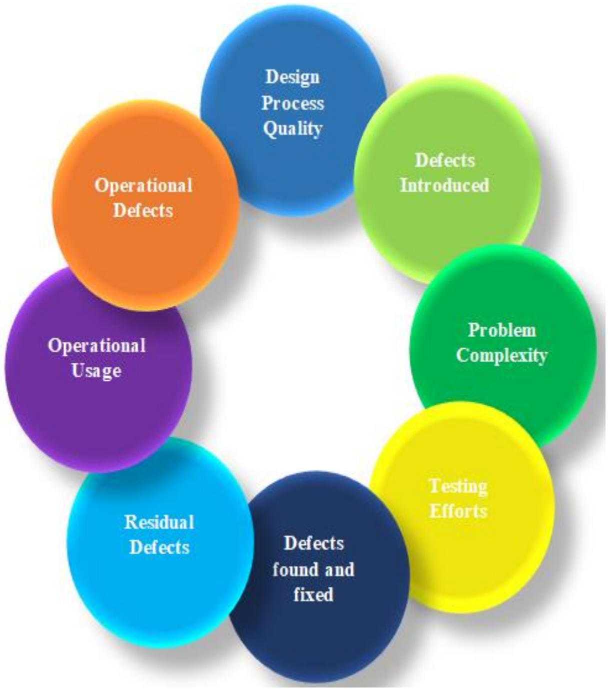


Fig9.

- Data mining and metrics are useful for establishing an empirical basis and without all of this effort; we could not have designed the models. So there is need of collaboration of data mining and metrics first. The true needs are answered by utilizing the kind of causal models we seen. These models enable us to address the kinds of questions that software project managers and developers need to address throughout their projects' lifetime. These models can help to prevent software project errors, as it allows managers to carry out proper quantified risk evaluation early and make more reasonable decisions based on software metrics [4], [5].
- Especially we need models that can handle the various processes and product evidence, uncertainty and incomplete information. I can suggest the example of BBN BBNs received more recent interest as a response to decision-making challenges aid under uncertainty. A BBN is a graphical network and a connected collection of probability tables [4].
- The Bayesian approach allows for methodological inferences Increase expert judgment in these areas problem domain in which empirical data is scanty. A PROBABILISTIC MODEL FOR DEFECT PREDICTION was build goal for this model is that to build prediction of module level defect model which could then be assessed against real project data. The purpose to add this model's example is that there should me more models in software metrics to make sure effective measurement with best software quality [19].

7 CONCLUSION

The metrics are used in software engineering to develop process efficiency and product effectiveness. Software metrics can help software professionals make software processes and products' unambiguous software attributes visible. Measurement often includes quantitative software tests, and metrics may typically be used specifically to quantitatively evaluate the outcomes of quality targets. The existing software metrics management is inefficient because of the incredibly complicated software development. So in this paper an attempt has been made to find out the challenges and then we tried to give some suggestions in the form of model which really help to overcome these challenges. The new research directions will help in further development of software metrics and provide a road map for advance and error free measurements. This work can be expanded further, focusing on software measurement and measurements principles and methods in software engineering.

REFERENCES

[1] Sharma, Reena, and R. S. Chhillar. "Novel Approach to Software Metrics." International Journal of Soft Computing and Engg 2.3 (2012): 232-236.
[2] Lincke, Rüdiger, Jonas Lundberg, and Welf Löwe. "Comparing software metrics tools." Proceedings of the 2008 international symposium on Software testing and analysis. 2008.
[3] Novak, Jernej, and Gordana Rakić. "Comparison of software metrics tools for: net." Proc. of 13th International Multiconference Information Society-IS. 2010.
[4] Fenton, Norman E., and Martin Neil. "Software metrics: successes, failures and new directions." Journal of Systems and Software 47.2-3 (1999): 149-157.
[5] Fenton, Norman, and Martin Neil. "New directions in software metrics." CIO Symposium on Software Best Practices. 2006.
[6] Dekkers, Carol A., and Patricia A. McQuaid. "The dangers of using software metrics to (mis) manage." IT professional 4.2 (2002): 24-30.
[7] Hall, Tracy, and Norman Fenton. "Implementing effective software metrics programs." IEEE software 14.2 (1997): 55-65.
[8] Grady, Robert B. "Successfully applying software metrics." Computer 27.9 (1994): 18-25.
[9] Patrick, Kua. (Online: Retrieved on 8 May 2020). "What Are Software Metrics and How Can You Track Them, https://stackify.com/track-software-metrics/#wpautbox_about"
[10] Hoffman, Doug. "The darker side of metrics." Pacific Northwest Software Quality Conference. Vol. 17. 2000.
[11] Kaner, Cem, James Bach, and Bret Pettichord. Lessons learned in software testing. John Wiley & Sons, 2008.
[12] Van Solingen, Rini, and Egon Berghout. "Integrating goal-oriented measurement in industrial software engineering: industrial experiences with and additions to the Goal/Question/Metric method (GQM)." Proceedings Seventh International Software Metrics Symposium. IEEE, 2001.
[13] Basili, Victor R., et al. "Linking software development and business strategy through measurement." Computer 43.4 (2010): 57-65.
[14] Basili, Victor, et al. "GQM+ Strategies: A comprehensive methodology for aligning business strategies with software measurement." arXiv preprint arXiv:1402.0292 (2014).
[15] Novak, Jernej, and Gordana Rakić. "Comparison of software metrics tools for: net." Proc. of 13th International Multiconference Information Society-IS. 2010.

[16] Rico, David F. "Practical metrics and models for return on investment." TickIT International 7.2 (2005): 10-16.
[17] Kadapa, Sanjana. "Challenges of Software Metrics Estimation." International Journal of Scientific and Research Publications 6.9 (2016): 883-889.
[18] Husein, Sukainah, and Alan Oxley. "A coupling and cohesion metrics suite for object-oriented software." 2009 International Conference on Computer Technology and Development. Vol. 1. IEEE, 2009.
[19] Krause, Paul, Bernd Freimut, and Witold Suryn. "New directions in measurement for software quality control." 10th International Workshop on Software Technology and Engineering Practice. IEEE, 2002.
[20] Srinivasan, K. P. "Unique Fundamentals of Software Measurement And Software Metrics In Software Engineering." International Journal of Computer Science & Information Technology (IJCSIT) 7.4 (2015).
[21] Chopra, Kunal, and Monika Sachdeva. "EVALUATION OF SOFTWARE METRICS FOR SOFTWARE PROJECTS." International journal of computers & technology 14.6 (2015): 5845-5853.