A Study on Software Metrics and Phase based Defect Removal Pattern Technique for Project Management

Jayanthi.R, M Lilly Florence

Abstract: ***The Software engineering is a application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software. Metrics are used by the software industry to quantify the development, operation and maintenance of software. They give us knowledge of the status of an attribute of the software and help us to evaluate it in an objective way. In software development, a metric is the measurement of a particular characteristic of a program's performance or efficiency. Basically, as applied to the software product, a software metric measures (or quantifies) a characteristic of the software. Metrics provide information to support quantitative managerial decision-making during the software lifecycle and also software metrics is a collective term used to describe the very wide range of activities concerned with measurement in software engineering. The practice of applying software metrics to a software process and to a software product is a complex task that requires study and discipline and which brings knowledge of the status of the process and / or product of software in regards to the goals to achieve Phase-based defect removal pattern.***

Keywords: Metrics,ModelsProject, Product, Quantitative

I. INTRODUCTION

Effective management of any process requires quantification, measurement, and modeling. Software metrics provide a quantitative basis for the development and validation of models of the software development process. Metrics can be used to improve software productivity and quality. Although current metrics and models are certainly inadequate, a number of organizations are achieving promising results through their use. Results should improve further as we gain additional experience with various metrics and models.

It has been noted frequently that we are experiencing a software crisis, characterized by our inability to produce correct, reliable software within budget and on time. No doubt, many of our failures are caused by the inherent complexity of the software development process, for which there often is no analytical description. These problems can be resolved, however, by improving our software management capabilities. This requires both the development of improved software metrics and improved utilization of such metrics.[1].

By evaluating an attribute of the software, we can know its status. From there, we can then identify and classify what its situation is; which helps us to find opportunities of improvements in the software.

Manuscript received September, 2013.

Jayanthi.R, MCA, VTU / PESIT- South Campus / Bangalore, India, India

Dr. M Lilly Florence, Department of MCA, Anna university/ Adhiyamaan engg College(ACE)/ Hosur,Tamilnadu, India,

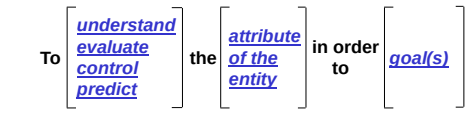
This also helps us to latter make plans for modifications that need to be implemented in the future. In addition, to save the values obtained as history for further reference. There is a broad range of software metrics. The software metrics depend on what software's attributes we want to quantify or qualify is based on **Phase-based defect removal pattern** for the software product.

II. TYPE OF SOFTWARE METRICS

Software metrics can be classified into three categories: product metrics, process metrics, and project metrics. product metrics describe the characteristics of the product such as size, complexity, design features, performance, and quality level. Process metrics can be used to improve software development and maintenance. Examples include the effectiveness of defect removal during development, the pattern of testing defect arrival, and the response time of the fix process. Project metrics describe the project characteristics and execution. Examples include the number of software developers, the staffing pattern over the life cycle of the software, cost, schedule, and productivity.

- *Measure* - quantitative indication of extent, amount, dimension, capacity, or size of some attribute of a product or process.
  - E.g., Number of errors
- *Metric* - quantitative measure of degree to which a system, component or process possesses a given attribute. "A handle or guess about a given attribute."
  - E.g., Number of errors found per person hours expended.

Metrics Objective Statement Template



Example - Metric: % defects corrected

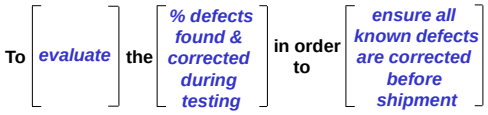


FIG.1

Software quality metrics are a subset of software metrics that focus on the quality aspects of the product, process, and project.

- *process* - used to develop the SW
- *project* - specific SW development project
- *product* - SW produced

many of the same metrics apply to both the process and project domains

Mathematically, a metric is a function m measuring the *distance between two objects* such that:

1. ∀ x, m(x,x) = 0
2. ∀ x, y, m(x,y) = m(y,x)
3. ∀ x, y, z, m(x,z) ≤ m(x,y) + m(y,z)
4. LOC - a function of complexity
5. Language and programmer dependent
6. Halstead's Software Science (entropy measures)
7. n_1 - number of distinct operators
8. n_2 - number of distinct operands
9. N_1 - total number of operators
10. N_2 - total number of operands

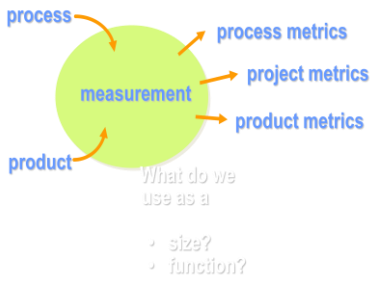


Fig.2

2.1 Products Metrics:

Product metrics are also known as quality metrics and is used to measure the properties of the software. Product metrics includes product non reliability metrics, functionality metrics, performance metrics, usability metrics, cost metrics, size metrics, complexity metrics and style metrics. Products metrics help in improving the quality of different system component & comparisons between existing systems.Product metrics is quantify characteristics of the product being developed-size, reliability

- ✓ Focus on the quality of deliverables
- ✓ Product metrics are combined across several projects to produce process metrics
- ✓ **Metrics for the product:**
- – Measures of the Analysis Model
- – Complexity of the Design Model
- – Internal algorithmic complexity
- – Architectural complexity
- – Data flow complexity
- – Code metrics

2.2 Process Metrics:

Process metrics are known as management metrics and used to measure the properties of the process which is used to obtain the software. Process metrics include the cost metrics, efforts metrics, advancement metrics and reuse metrics. Process metrics help in predicting the size of final system & determining whether a project is running according to the schedule.

Process metrics -quantify characteristics of the process being used to develop the efficient software by detecting fault.

Reliability:-The probability that a program will perform its specified function, for a stated time period, under specified conditions.

- ❖ MTBF (Mean Time between Failures)
- ❖ MTTF (Mean Time to Failures)

Advantages:

1. Short, quick, simple and easy.
2. When answered properly for all questions then this method has the highest tested accuracy.

Disadvantages:

1. Quick and easy one parameter models can also mean less accuracy.
2. Can not apply to every project.

2.3. Project metrics:

A software team can use software project metrics to adapt project workflow and technical activities. Project metrics are used to avoid development schedule delays, to mitigate potential risks, and to assess product quality on an on-going basis. Every project should measure its inputs (resources), outputs (deliverables), and results (effectiveness of deliverables).

III. SOFTWARE METRICS MESURING ATTRIBUTES AND MAPPINGFOR PROJECT MANAGEMENT

Software development managers are responsible for the timely completion of projects. They try to effectively focus team effort on the appropriate activities to complete projects on schedule and with high quality. In order to judge the status of projects so that teams can react accordingly, managers need project measurements which consist of both product and process metrics.

3.1.Types of Measures

Direct Measures

- > *Measured* directly in terms of the observed attribute (usually by counting)
- — Length of source-code, Duration of process, Number of defects discovered
- > Direct Measures (internal attributes)
- — Cost, effort, LOC, speed, memory

Indirect Measures

- > *Calculated* from other direct and indirect measures
- — Module Defect Density = Number of defects discovered / Length of source
- — Temperature (usually derived from the length of a liquid column)
- > Indirect Measures (external attributes)
- — Functionality, quality, complexity, efficiency, reliability, maintainability

3.2.Design Complexity: Referred as structural feature of design artifacts that can be measured before the implementation phase and involves the modeling of information flow in the application.

3.3.Design stability: Can be measured at any point during the design process and is the quality attribute which represents the resistance to the potential ripple effect which a program developed from some design will have when it is modifiedEvery company pay special attention to its work process management and for that they always look for tools that could help them in this endeavor. Now the companies search has come to end with the introduction of practical software metrics for project management and process improvement software. It helps in easily managing the work process as well as managing the projects. It empowers organizations to control the capture and processing of form based information required for critical business process automation and project handling. It

replaces inefficient paper based versions with electronic form based workflows and provides automatic workflows that are perceptible and web-based, increasing productivity, ensuring compliance and accountability as well as improving efficiency.[2] Practical software metrics for project management and process improvement helps in managing and implementing even the most complex workflow and ensures the process automation obtained precisely reflects the business needs, including access to and decisions based on third party applications.

Practical software metrics for project management and process improvement offers features like data verification in forms to maintain data quality in an automated process and to ensure that the information collected is correct and valid for business process and also helps in filtering and reporting any business process according to specific requirements . Practical software metrics for project management and process improvement also helps in workflow automation that helps us to define the way the data should be displayed, any constraints that should be applied, and the way it should react if changed, as each field is configurable. All these features offered by it helps companies in easily managing their projects as well as their work process.

3.4.Typical Size-Oriented Metrics

- errors per KLOC (thousand lines of code)
- defects per KLOC
- $ per LOC
- page of documentation per KLOC
- errors / person-month
- LOC per person-month

We can calculate the following Size Oriented Metrics from the table or each Projects: e.g. for Project A

Errors / KLOC (134 / 12.1 = 11.07
Defect / KLOC (29 / 12.1 = 2.4
$ / KLOC (168000 / 12.1= 13884)
Page of Document / KLOC (365 / 12.1 = 31.7)

Note: KLOC (Thousand of Lines of Code. (eg 12100 LOC = 12.1 KLOC)

Other interesting Metrics can be computed such as:-

- **Errors / Person-Month (134 / 24 = 5.58)**
- **LOC / Person-Month (12100 / 24 = 504)**
- **$ / Pages of Documentation (168000 / 365 = 460.27)**

Validity and reliability

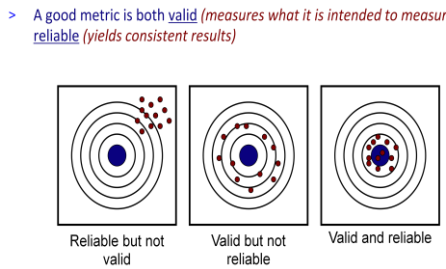


FIG.3

IV.THE IMPORTANCE OF METRICS TO SOFTWARE PROJECTS

- When measurements embrace a structure or system they become more meaningful indicators called metrics. The structure could be a simple algebraic formulation or model. The structure behind metrics could be built on software engineering problems or business situations. Moving from measurements to metrics is like moving from observation to understanding. Metrics are conceived by the user and designed to reveal a chosen characteristic in a reliable and meaningful manner. [3].Metrics are also indicators. Each metric becomes a natural element in the organization's business intelligence system or management information system. A well – structured metrics process would help in taking data driven decisions in time. In the broadest terms, the objectives of software measurement are to allow managers and practitioners to make timely, data-driven decisions; to track an organization's progress toward its improvement goals; and to assess the impact of process changes. In addition to using software metrics to measure specific attributes of a software product, process, or resource, the project manager can use them to analyze :-
- product errors and defects
- Assess status
- Derive a basis for estimates
- Determine product complexity
- Establish baselines
- Experimentally validate best practices
- Predict quality, schedule, effort, and cost
- Track project progress
- Understand when a desired state of quality, in a product or in a process, has been achieved.

Software metrics are used to obtain objective reproducible measurements that can be useful for quality assurance, performance, debugging, management, and estimating costs. Finding defects in code (post release and prior to release), predicting defective code, predicting project success, and predicting project risk. There is still some debate around which metrics matter and what they mean, the utility of metrics is limited to quantifying one of the following goals :

- Schedule of a software project,
- Size/complexity of development involved,
- cost of project, quality of software.
- Estimate the cost & schedule of future projects.
- Evaluate the productivity impacts of new tools and techniques.
- Establish productivity trends over time.
- Improve software quality.
- Forecast future staffing needs.

3.5.Measurement Mapping

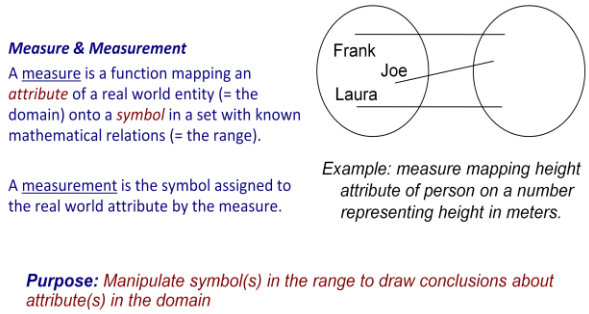


Fig.4

Despite the fact that the Software Engineering field doesn't have a unified set of metrics that the community has agreed to use, it is advised to use them. Often during the software process, the members of the development team do not know if what they are doing is correct and they need a guide that could help them orientate further improvement and to objectively know if the improvement is being achieved. Software metrics are tools that

help to track software improvement.
Most large companies dedicated to develop software, use metrics in a consistent way. Many companies have created their own standards of software measurement; so, the way that metrics are applied usually varies from one company to another one. Nevertheless, as they are used in a consistently way through different projects, the software groups get many benefits from them.
What to measure in regards of software process or product depends on the nature of the project, but in all cases, the customer satisfaction is the goal and measures should be taken to achieve that goal not only at delivery, but through the entire development process. There are a wide range of software metrics. we apply these metrics to a sample project, and evaluate the results. We find that that there are specific metrics for different stages of the software development cycle. When used properly, i.e., when a company uses the best software metric during each development phase, the quality of the software will dramatically increase. Therefore, we highly recommend using software metrics during all stages of the development process.

IV. RECENT STUDIES

The field of software metrics has sufficiently matured so as to allow project managers and software engineers to use metrics to tune software processes rather than focus on software products. Metrics are now used in throughout the software life-cycle, from specification to maintenance. While traditional uses still exist, for example a development organization may establish a threshold for complexity above which source code is required to be re-engineered, more recent efforts focus on using metrics for high level project management decisions. The Mitre Corporation has used software metrics in management decisions such as when to integrate subsystems, evaluating test schedules, choosing between maintaining or redesigning systems, and determining when projects are no longer on schedule and thus need to be re-planned [4]. Metrics can also be used to evaluate the status of a development project based on **Phase-based defect removal pattern .**

4.1.Phase-based defect removal pattern .

- A simple metric: ***defects/KLOC*** or ***defects/FP***
- Defect rate during formal machine testing is usually connected with the defect rate in the field
- Higher defect rates found during testing indicate that the software has experienced high error injection – unless some new testing approach is used or some extra testing is used for some reason
- Indicates the quality of the product when the software is still tested – many defects in testing mean that there's too much error input in process can be used in other metrics.
- Phase defect removal effectiveness and related metrics are useful for quality management and planning
- Measurements that indicate clearly which phase of the process needs improvement and should be focused on
- Using the right tools, analyses can be done for the entire project as well as for local areas

4.2.Phase based Defect/ Error prediction Metrics

Provide the approximation of total number of errors that can be found during the entire development process. External *measure.-Me*asure the number of defects/ errors in a software product and data required for the metrics is collected from the product itself.
Advantage: Helps to find error during development process.
Disadvantage: There is no particular way to measure defects**.**

4.3.Defect Removal Efficiency

Defect removal efficiency provides benefits at both the project and process level. It is a measure of the filtering ability of QA activities as they are applied throughout all process framework activities
It indicates the percentage of software errors found before software release
It is defined as DRE = E / (E + D)
E is the number of errors found before delivery of the software to the end user
D is the number of defects found after delivery
As D increases, DRE decreases (i.e., becomes a smaller and smaller fraction)
The ideal value of DRE is 1, which means no defects are found after delivery
DRE encourages a software team to institute techniques for finding as many errors as possible before delivery

Defect Removal Effectiveness (DRE)

A simple metric:

$$DRE = \frac{\text{Defects removed during a development phase}}{\text{Defects still hidden in the product}} \times 100\%$$

Of course the number of latent defects in the product at any given phase is not known, so the metric is based on approximation. Its usually estimated by:

Defects removed during the phase + defects found later

Fig.5

4.4.Pattern: Study the Exceptional Entities

***Problem*-How can you quickly gain insight into complex software?**
Solution*-Measure*** software entities and *study the anomalous one*
***Steps*-**Use simple metrics-Visualize metrics to get an overview- Browse the code to get insight into the anomalies

4.5.The Pattern of Testing Defect Arrival

Measuring the pattern of defect arrivals (times between failures).Objective is to look for defect arrivals that stabilize at very low level or times between failures that are very far apart before ending the testing.Indicates how the testing is done (if the patterns still occur as high level, badly) and future reliability.

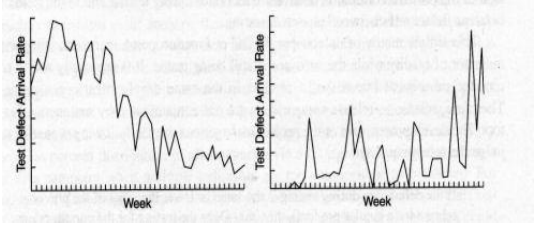


Fig.6

In this report we show the effective application of a small set of metrics, identified in a series of empirical studies, to assist managers in tracking and controlling software projects. This study was intended to identify metrics and/or other work artifacts useful for monitoring project status. A list of potentially useful metrics were selected with the intent of providing visibility across the major phases of the life cycle, from requirements specification through delivery [3].

V.CONCLUSION

Phase-based defect removal pattern metrics can also be used to evaluate the status of a development project. Software metrics can reveal a lot of information about the code at several stages of development.

They can identify the routines which need to be redesigned due to higher complexity, routines which may require thorough testing, and features which may require more support.

Measuring attributes of a software product under construction give insight to the stability and fitness for testing and delivery. Software measurement can also identify potential risks to a project Early risk identification provides a development team with an opportunity to focus effort on the critical project tasks .

The management metrics will certainly reflect the management system we settle for. The system provides sustenance, meaning and context. Metrics provide supportive definitions, Visibility and feel. When metrics operate and data flows in, imperfections in the structure become visible, leading to additional improvements.

REFERENCES

[1] G. Atkinson, An Analysis of Process and Product Metrics for Monitoring Software Project Status, Masters Thesis, Software Engineering Testing Laboratory, Univ. of Idaho, May 1997.
[2] V. R. Baislli & D. M. Weiss, “A Methodology for Collecting Valid Software Engineering Data”, IEEE Transactions of Software Engineering, SE-10(6), November 1984, pp. 728-738.
[3] V. R. Basili, G. Caldiera, and H. D. Rombach, The Goal Question Metric Approach, http://www.cs.calpoly.edu/~ssenkere/metric.html
[4] Stephen H. Kan, “Metrics and Models in Software Quality Engineering”, Pearson Education Limited 2003, Boston, United States

Jayanthi.R received a Master’s Degree in Computer Applications in 1998 in vellalar college for women, Erode(DT)and received a M.Phil(CS) in 2008 in Missions university,Salem.Doing PhD Part-Time in Bharathiyar University,Coimbatore.Also working as an Assistant Professor in Department of MCA in PESIT ,Bangalore South Campus. Research interests include Software engineering, object oriented Modeling and Design.

Dr. M. Lilly Florence working as a Professor in Department of MCA in Adhiyamaan Engineering College(ACE). Guiding research scholars in various discipline. Research interests include Software Reliability, software engineering