Quantitative Quality Management through Defect Prediction and Statistical Process Control[T]

Pankaj Jalote, K. Dinesh, S. Raghavan, M. R. Bhashyam, M. Ramakrishnan
Infosys Technologies Ltd.

Summary:
To produce high quality software, the final software should have as few defects as possible. The task of quality management in a software project is to plan suitable quality control activities, and properly execute and control these activities such that most of the defects are detected "in-process", that is, before the software is delivered. In this article, we describe the approach of quantitative quality management through defect prediction and statistical process control that is employed at Infosys, a large ISO-certified software house that has been assessed to be at level 5 of the CMM. In this approach, a quality goal is set for a project in terms of the defect density delivered. To achieve the goal, the defect levels for different phases in the process are estimated using past data. During the execution of the project, the actual defect numbers are compared with the estimates to see if the project is progressing satisfactorily towards achieving the goal, or some correction is needed. To further improve the control and provide early warnings, the phase-wise control is complemented with activity-level control using statistical process control.

Pankaj Jalote, Professor, Department of Computer Science and Engineering; Indian Institute of Technology, Kanpur – 208016, India; Email: jalote@iitk.ac.in, Fax: +91-512-590725/590413

K. Dinesh, S. Raghavan, M. R. Bhashyam, S. Ramakrishnan; Infosys Technologies Ltd.; Electronics City, Hosur Road, Bangalore – 561 229 Email: [kdinesh, raghavs, bhashyam, ramakrishnanm]@inf.com, Fax: +91-80-852-0362

1. Introduction

Quality, along with cost and schedule, is one of the major factors determining the success of a project. The concept of software quality is not as easily definable. There are various possible quality characteristics of software, and there is even an international standard for this [2]. In practice, frequently, quality management revolves around defects, and *delivered defect density*, that is, number of defects per unit size in the delivered software, has become the current de-facto industry standard [1].

To produce high quality software, the final software should have as few defects as possible. As software development is a human activity, it is not possible to prevent injection of defects (though it is possible to reduce the level of injection). Once it is accepted that defects will get injected and the objective is to deliver software with few defects, it is imperative to remove defects before delivery. Due to this, a project's process has many quality control activities for defect identification and removal. The task of quality management is to plan suitable quality control activities, and properly execute and control these activities such that most of the defects are detected "in-process", that is, before the software is delivered.

[T] Parts of this article have been adapted from portions of "CMM in Practice: Processes for Executing Software Projects at Infosys," by P. Jalote, copyright ©2000 by Addison Wesley Longman, Inc. Material used with permission of the publisher.

There are two approaches to quality management – the procedural approach and the quantitative approach. In the procedural approach for quality management, procedures and guidelines for the review and testing activities are established. In a project, these activities are planned (i.e. which activity will be performed and when), and during execution, they are executed following the defined procedures. The procedural approach to defect removal does not allow claims to be made about what percentage of defects have been removed, or the quality of the software after performing the procedure. It also does not provide quantitative means for a project manager to assess and manage the quality of the software being produced – the only factor visible to the manager is whether the quality control tasks are executed or not.

A quantitative approach goes beyond asking "has the method been executed" and looks at defect data to evaluate the effectiveness of the quality control activities and whether more testing or reviews need to be done. There are two key aspects to quantitative quality management – setting a quantitative quality goal, and then managing the software development process quantitatively so that the quality goal is met. Managing the process quantitatively requires that intermediate goals are set for different stages in the project, which if met during the actual project execution, will ensure that the quality goal is met. These intermediate goals can then be used for quantitatively monitoring the execution of the project. Whereas procedural approach to quality management is expected from a level 3 organization by the Capability Maturity Model [6], proper quantitative quality management is what is expected at level 4 of the CMM.

In this article we briefly describe an approach for quantitative quality management using defect prediction and statistical process control (SPC). This approach is the one that is used by Infosys, a large software house with headquarters in Bangalore, which is ISO certified and has been assessed at level 5 of the CMM. Further details about the approach can be found in [3].

2. Quantitative Quality Management

A good quality management approach should provide warning signs early in the project and not only towards the end, when the options available are limited. Early warnings will allow timely intervention. For this, it is essential to predict values of some parameters at different stages in the project such that controlling these parameters in project execution will ensure that the final product has the desired quality. If these predictions can be made, then the actual data during the execution of the process can be used to judge whether the process has been effectively applied. With this approach, a defect detection process does not finish by the declaration that the process has been executed – the data from process execution is used to ensure that the process has been performed in a manner that its full potential has been exploited.

Software reliability models [4, 5] or the concept of Defect Removal Efficiency [4] can be used for quantitative management of quality, though these measures have some limitations for quality management [3]. Infosys implements quantitative quality management through defect prediction. In this approach, the quality goal is set in terms of delivered defect density. Intermediate goals are set by estimating the defects that may be detected by various defect detection. In other words, once the quality goal has been set, the defect levels at different stages are estimated such that if the estimates are met then the target quality will be achieved. Then for process management, the predicted defect levels become the benchmark against which actual defect levels are compared to evaluate if the development process is moving in the direction of achieving the quality goal.

The effectiveness of this approach depends on how well can we predict the defect levels at different stages of the project. At Infosys, defect patterns observed in past projects are used for predicting defect levels.

Through this approach, phase-wise control can be established. However, this level of control is too "macro" for a project as a phase is too large an activity, and a finer or more "micro" control is needed such that corrective and preventive actions can be taken quickly. This is achieved by employing SPC technique to the two quality control activities that detect the maximum defects – reviews and unit testing. For employing SPC, based on past data, control limits are established for key parameters like defect density, coverage rate, etc.

We now further discuss how quantitative quality management is done at Infosys. At Infosys, past data is encapsulated in the Process Database (PDB) and the Process Capability Baseline (PCB) [3]. The former gives the summary data for each project, while the latter gives the average and ranges for various process characteristics. From past data, the defect injection rate, defect removal efficiency, etc. are all known.

Infosys has been collecting defect and effort data for over six years. Collecting and reporting data is part of the work culture and even the Directors and the CEO fill the on-line weekly timesheets to report where their effort went. Tool support is also provided for recording the two key measures – effort and defect (for effort recording, an in-house tool is used and for defect logging and tracking a commercial tool is used.) The PDB has now over 250 data points for projects completed over the last few years.

2.1 Setting the Quality Goal

The quality goal for a project is the number of defects the project expects to deliver. That is, the quality goal is the expected number of defects during acceptance testing. There are two primary sources of past data which may be used for setting the quality goal – past data on similar projects, and data from the PCB.

If data of similar projects is used, then the number of defects found during acceptance testing of this project can be estimated as the product of number of defects found during acceptance testing of the similar projects and the ratio of the estimated effort for this project and the total effort of the similar projects. If the set of similar projects whose data is being used is SP, and the current project is P, then the estimate for defects in acceptance testing of P is:

$$\text{Estimate for AT defects (P)} = \frac{\text{AT defects (SP)} * \text{Effort estimate (P)}}{\text{Actual effort (SP)}}$$

This gives the number of defects that can be expected if the same process is used as in the similar projects. If data from the process capability baseline is used, then there are a few different ways to compute this.

A project can set a quality goal that is better (or worse) than the quality level of a similar project or what is expected from the standard process. If this is done, then the expected number of defects for the higher goal can be determined as above by using the quality goal set for the project. In this case, the process used by the similar project or the standard process will have to be modified for use in this project to ensure that the quality target is met.

2.2 Estimating Defects for Other Stages

Once the process has been designed to meet the quality goal of the project, defect levels for the different quality control activities needs to be estimated for quantitatively controlling the quality. The approach for estimating defect levels for some defect detection activity is similar to the approach for estimating the defects in acceptance testing. From the estimate of the total number of defects that will be injected during the course of the project, the defect levels for different testing stages are estimated by using the percentage distribution of defects as given in the PCB (or as reported for similar projects in the PDB). The percentage distributions that are given in the PCB are:

Process Stage	**% of Total Defects**
Req Spec Review + HLD Review + Detail Design Review	15 - 20 %
Code reviews + Unit Testing	50 - 70 %
Integration Testing + System	20 - 28 %

Testing	
Acceptance Testing	5 - 10 %

If the quality goal has been set based on the data from some similar projects, and the quality goal is higher than that of the similar projects, it is unreasonable to expect that by following the same process as the earlier projects did the higher quality goal will be achieved. If the same process is followed, the reasonable expectation is that similar quality levels will be achieved. Hence, if a higher quality level is desired, the process has to be suitably "upgraded". Similarly, if the quality goal is set to be higher than the quality levels given in the PCB, then it is unreasonable to expect that by following the standard process the higher quality level will be achieved. Hence, a strategy will be needed to achieve the higher quality goal. This strategy is generally a combination of training, prototyping, testing, and reviews, and is explicitly stated in the project management plan for the project.

The data in the process-capability baseline can be used to estimate the impact of process changes on the effort and schedule planning of the project. Once the process is decided, the defect levels at the different stages of the process can also be estimated. Using data on effort per defect, which can be obtained from the effort distribution data and the defect distribution data, the impact on effort of different activities in the process can be estimated. Though it is possible to estimate the impact of changes on the effort and schedule by using the past data, quite likely, once the process changes are identified, impact of these changes is estimated based on past experience. This is usually acceptable, as the changes are generally minor.

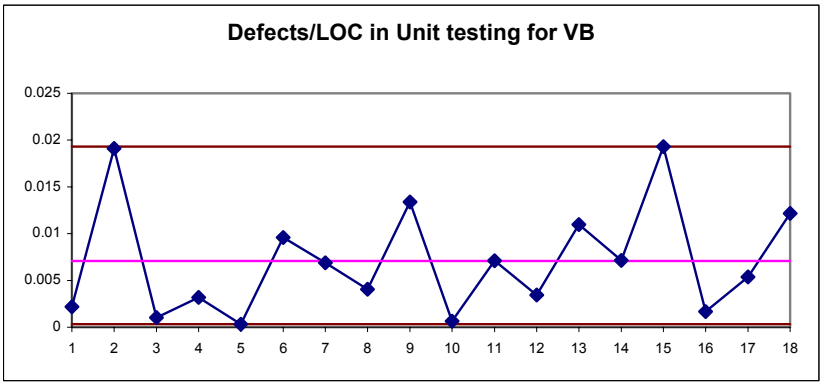
2.3 Micro Control through SPC

The method discussed above provides a phase-wise control. If the quality control activities of a phase do not perform well in a project, then this fact can be detected at the end of the phase. Based on this information, the project manager can take corrective and preventive actions for the project. However, the phase-wise control is too macro as it allows evaluation only at the end of phases. It is desirable if actions can be taken even during the phase such that the phase-wise goals are met. This is achieved by employing SPC.

At Infosys, micro-level SPC is mostly employed for the quality control activities relating to the coding phase, as this is the phase in which most defects are detected. The two main quality control activities of this phase are code reviews and unit testing. For both of these, SPC is employed to evaluate the execution of each review and unit testing.

For reviews, based on past data, control charts are built for different types of reviews. From the control charts, the review capability baseline is established, which gives the control limits for key parameters like defect density found in reviews, preparation rate, review rate, etc. Setting of these limits is discussed in [3]. These limits are used to evaluate a review. At the end of each review, it is checked if the defect rate is within the limits. If yes, nothing needs to be done. If no, then other rates are checked, and based on evaluation, some corrective and preventive actions may be taken. Guidelines have been provided for evaluation. These guidelines can be found in [3].

For unit testing, control charts are built, mostly for defect density. A control chart for one of the languages is shown below:



The results of a unit testing are checked against the control limits. Again, action may be taken if the results are out of the limit, and guidelines have been provided for evaluation [3].

SPC is largely accepted in the company and most projects use it to track defects found and effort consumed in quality control tasks. To help the project managers apply SPC, an in-house built tool has been provided to employ SPC on a project. The SPC tool does not employ company-wide control limits throughout the duration of the project but allow it to change depending on the actual defect data. What has been observed that frequently in the start the parameters are sometime out of control limits but with attention being drawn to these by the SPC tool, as the project progresses, they reach acceptable levels. Also, the SPC tool uses, besides the past performance data, specification limits which are set based on the improvement goals of the organization.

3. Summary

To produce high quality software, the final software should have as few defects as possible. The task of quality management is to plan suitable quality control activities, and properly execute and control these activities such that most of the defects are detected "in-process", that is, before the software is delivered. There are two approaches to quality management – the procedural approach and the quantitative approach. In the procedural approach for quality management, procedures and guidelines for the review and testing activities are established. In a project, these activities are planned and executed.

A quantitative approach to quality management goes beyond asking "has the method been executed" and looks at defect data to evaluate the effectiveness of the quality control activities to achieve some quantitative quality goals. There are two key aspects to quantitative quality management – setting a quantitative quality goal, and then managing the software development process quantitatively so that the quality goal is met. Managing the process quantitatively requires that intermediate goals are set for different stages in the project, which if met during the actual project execution, will ensure that the quality goal is met. These intermediate goals can then be used for quantitatively monitoring the execution of the project. Whereas procedural approach to quality management is expected from a level 3 organization by the CMM, proper quantitative quality management is what is expected at level 4 of the CMM

We discussed the approach of quantitative quality management through defect prediction and statistical process control that is employed at Infosys, a large software house that has been assessed to be at Level 5 of the CMM. In this approach, based on current process capability and process improvement objectives, a quality goal is set in terms of the defect density delivered. To achieve the goal, the defect levels for different phases in the process are estimated using past data. During the execution of the project, the actual defect numbers are compared with the estimates to see if the project is progressing satisfactorily towards achieving the, or some correction is needed.

To further improve the control and provide warnings early, the macro-level, phase-wise control is complemented with activity-level control using statistical process control. For activities like reviews and unit testing, based on past data, control charts have been built for key characteristics like defect density, coverage rate, etc. As each review or unit testing is completed, the results are compared against the control limits. Corrective and preventive actions may be taken if the data is out of limits.

However, like most performance metrics in software, since large variations are possible, when the actual number of defects or the defect density is lesser than the "target", one cannot say whether the injection rate is low or the removal process was not executed properly. And to resolve this situation, the project will have to look at other indicators [4]. Keeping this uncertainty in mind, if the actual data is "out of range", this fact is used to draw attention of the project management, which will look at other indicators, like effort spent, to decide what is the actual situation and what action, if any, needs to be taken.

At Infosys, these procedures are now standard practice and defect predication and comparison with actual and SPC are used in all projects. The details of use, however, may differ in different projects. For example, some projects may predict defect levels for different phases. Similarly, different projects use SPC for controlling different tasks – some use it primarily for reviews while some use it for unit testing only, while some use it to monitor the arrival rate of requests or percentage deviation in effort or schedule.

Though in this report we have focussed mostly on monitoring of quality control activities, for ensuring high quality other techniques are also used. Defect prevention is one such technique that is used at Infosys, and is required for a level 5 organization. In defect prevention, patterns in past defects, both at the organization level and at the project level, are used to prevent defects from getting injected. In other words, whereas defect control focuses on removing the defects that have been injected, defect prevention focuses on reducing the injection of defects themselves. Good defect prevention techniques coupled with good techniques for executing and monitoring quality control activities provide a powerful combination for delivering high quality software.

References:

[1] N. E. Fenton and S. L. Pfleeger, *Software Metrics, A Rigorous and Practical Approach,* Second Edition, International Thomson Computer Press, 1996.
[2] International Standards Organization, *Information Technology – Software Product Evaluation – Quality Characteristics and Guidelines for their Use,* ISO/IEC IS 9126, Geneva, 1991.
[3] P. Jalote, *CMM in Practice – Processes for executing software projects at Infosys,* Addison Wesley Longman, SEI Series on Software Engineering, 1999.
[4] S. H. Kan, *Metrics and Models in Software Quality Engineering,* Addison-Wesley, 1995.
[5] J. D. Musa, A. Iannino, and K.Okumoto. *Software reliability---measurement, prediction, application*. McGraw Hill Book Company, 1987.
[6] M. Paulk et. al. *The Capability Maturity Model: Guidelines for Improving the Software Process,* Addison Wesley, 1995.