**TELKOMNIKA**, Vol. 11, No. 8, August 2013, pp. 4609~4615
e-ISSN: 2087-278X



# Study on Software Quality Improvement based on Rayleigh Model and PDCA Model

**Ning Jingfeng*, Hu Ming**
College of Computer Science and Engineering, Changchun University of Technology
Changchun 130012, China
*Corresponding author, e-mail: ningjingfeng@mail.ccut.edu.cn, huming_ccut@163.com

**Abstract**

*As the software industry gradually becomes mature, software quality is regarded as the life of a software enterprise. This article discusses how to improve the quality of software, applies Rayleigh model and PDCA model to the software quality management, combines with the defect removal effectiveness index, exerts PDCA model to solve the problem of quality management objectives when using the Rayleigh model in bidirectional quality improvement strategies of software quality management, and puts it into the application to achieve good results.*

***Keywords:*** *quality management, rayleigh model, PDCA model, defect analysis*



## 1. Introduction

With continuous advance of the economic globalization, the era of knowledge economy has come. Software has been the rapid rise as an emerging high-tech industry [1]. The rapid development of information technology enables software to be applied to various social fields, and software quality becomes the focus accordingly [2]. Dr. Joseph M. Juran, a famous American quality management scientist, pointed out "The 21st century is the century of quality; quality is the most effective weapon for the peaceful occupation of the market". There is no exception for the software industry's gradual maturity, software quality is being considered as the life of the software industry. Software quality management has developed in an all round way in the software organization, strong quality consciousness is gradually taking root in the hearts of the software technical and management personnel, till the forming of the whole organization quality culture. The article combines using reliability model-Rayleigh model and quality improvement model-PDCA model, on the basis of using the existed data, analyzes the defect, monitors and evaluates the software's quality, and gives judgment basis on whether the product can be released or not.

## 2. Rayleigh Model

Rayleigh model is a common reliability model that can predicate the defects distribution of the whole life cycle of software development [3]. Rayleigh model is a member of the Weibull distribution family. It has been several decades since Weibull [4] distribution used in reliability analysis in different engineering fields, which is one of the three famous extreme value distributions (Tobias, 1986). One of its symbolic features is that the tail of its probability approaches zero gradually, but cannot reach zero. In 1982, Trachtenberg [5] observed the monthly defect data of a set of software projects, and found that the comprehensive defect mode of the projects meet Rayleigh curve. In 1982, Gaffney of IBM Federal Systems Department reported, the time distribution of the software life cycle of defect founded in the 6 public defect detecting phases used by IBM, along with theses phases also meet Rayleigh curve. Its cumula distribution function (CDF) and probability density function (PDF) are as follows:

CDF: $F(t) = 1 - e^{-(t/c)^m}$ PDF: $f(t) = \frac{m}{t}\left(\frac{t}{c}\right)^m e^{-(t/c)^m}$

*Received March 7, 2013; Revised May 18, 2013; Accepted May 29, 2013*

Where m is the shape parameter, c is the scale parameter, and t is time. When applying to software, PDF often indicates the defect density (rate) changes over time or defect occurrence mode (defective data), while CDF indicates the occurrence mode of cumulative defect.

In Weibull family, the two models already used in software reliability are the models of shape parameters m=1 and m=2. Rayleigh model is the special case of Weibull distribution m=2, where CDF and PDF are as follows:

CDF: $F(t) = 1 - e^{-(t/c)^2}$ PDF: $f(t) = \frac{2}{t}\left(\frac{t}{c}\right)^2 e^{-(t/c)^2}$

PDF of Rayleigh first increases to peak value, then decreases at decreasing rate. Parameter c is the function of $t_m$, and $t_m$ is the time that the curve reaches the peak value. Take derivative of t from f (t), and make it be zero, solve the simultaneous equation to get $t_m$.

$$t_m = \frac{c}{\sqrt{2}}$$

After $t_m$ is estimated, the shape of the whole curves can be determined. The area of $t_m$ portion below the curve is 39.35% of the total area.

The above formula indicates standard distribution; in particular, the total area below the PDF curve is 1. In practical application, the formula is multiplied by the constant K (K is the total number of defects or the total cumulative defect rate). If we still make substitution in the formula,

$$C = t_m\sqrt{2}$$

We'll get the following formula. In order to determine model from one data point set, K and $t_m$ are parameters to be estimated.

$$F(\mathrm{t}) = K[1 - e^{1(1/2t^2{}_m)t^2}] \qquad f(t) = K\left[\left(\frac{1}{\mathrm{t}_m}\right)^2 te^{-(1/2t^2{}_\mathrm{m})t^2}\right]$$

Rayleigh model involves contents such as defect prevention and prophase defect removal related to the projects in early phase. On the basis of this model, if reducing the filling rate of error, the area blow the Rayleigh cure will become smaller, resulting in a smaller prediction field defect rate, as shown in Figure 1:

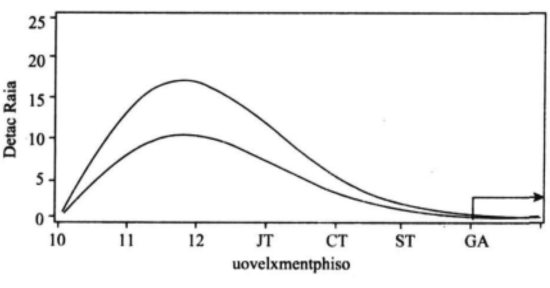


Figure 1. Rayleigh Model I

Similarly, if more defects are removed in the early phase of development process, the defect rate will be lower in the later phase of testing and maintenance, as shown in Figure 2.

Figure 3 describes the strategy to make quality improvement from two directions. From Figure 3, we can see our current quality improvement target is to reduce the height of the curves as much as possible, meanwhile move the crest of the Rayleigh curves to the left.

Among them, I0: high level design review; I1: low level design review; I2: code inspection; UT: unit test; CT: component test; ST: system test.

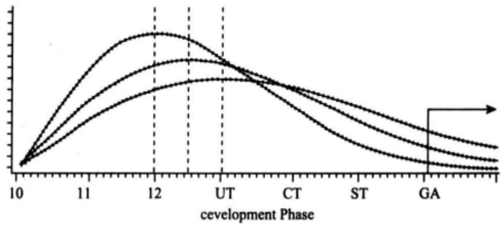


Figure 2. Rayleigh Model II

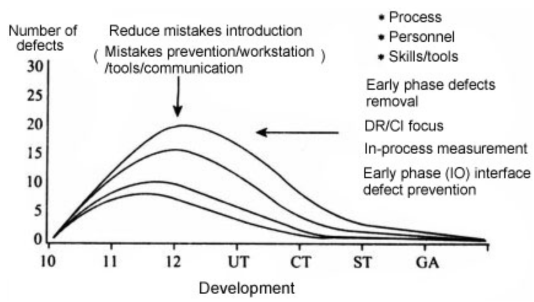


Figure 3. Directional Diagram of Development Quality Improvement

## 3. PDCA Model Analysis

PDCA is the acronym for Plan, Do, Check and Action. PDCA is also called Deming cycle, which is a classic quality management model promoted and practiced in Japan by Dr. W. Edwards Deming, an American quality management expert, as shown in Figure 4.

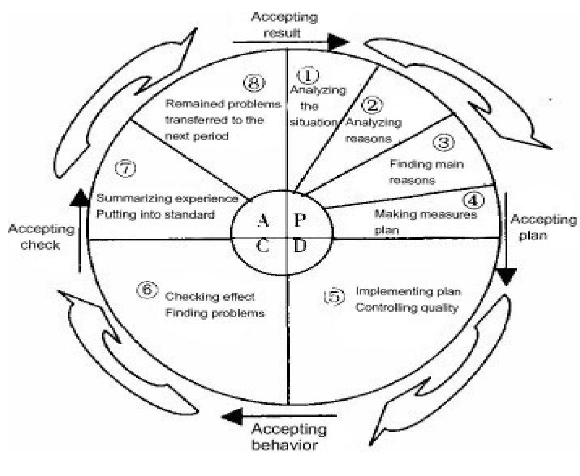


Figure 4. PDCA Cycle Process

Deming cycle is used as a model to make continuous process improvement using CMM by SEI, which was called IDEAL (Initiating, Diagnosing, Leveraging) [7]. ISO9001: 2000 stated in its introduction that: PDCA method is available for all processes. While all products are the results of process, the products quality is related to the process of setting up the products [8]. The improved PDCA theory has been widely used in quality management of the enterprise.

Meanwhile, PDCA also becomes a logical working process that enables any effective activities [9].

### 3.1. Plan

It includes the decision of principles and aims, and the formulation of activities process. The plan requires "5W1H", that is, what, why who, when, where and how [8], find problems and find out the reasons and the main reasons, set up quality principles, aims, letter of intention and management principles, etc. For example, the management principles have "processing methods", "management system methods" and "continuous improvements", etc.

### 3.2. Do

The second phase "Do" is not the simple "do". Do is the implementation and practice of the plan, and it is mainly to do according to the plan, to implement the practical measures, and control the process, enabling the activities to go forward as expectation and finally reach the plan and the target set. The implement of measures shall include 3 parts of contents: Do, control and regulate.

### 3.3. Check

Check is an evaluation for the effect after implementation. Check is accompanied the implementation process from beginning to end, it is the process of continuously collecting data, and getting information, and complete the check by data analysis and results measurement. Check shall undergo sufficient planning even at the beginning of the implementation, so as to make good evaluation for the results. Internal audit is a major check.

### 3.4. Action

The key lies in that the measures shall be taken after checking the results, i.e., to sum up successful experience and learn from failures, to implement standardization as basis for the future. Action is the sublimation process of the PDCA cycle. Without action, there is no improvement.

As the basic method of quality management, PDCA cycle is not only suitable for the whole software engineering, but also for the whole software enterprises and each department and even individual in the software enterprises. Each department has its own PDCA cycle according to the policy aim of the software enterprise, cycling layer upon layer, in the form of big ring linking with small ring, and small ring linking with smaller ring. Big ring is the matrix and basis of the small ring, while small ring is the decomposition and guarantee of big ring. PDCA cycle is like climbing stairs, when finishing one cycle, the quality of the production will improve one step, then setting up the next cycle, rerunning and reimproving, thus going forward and improving continuously, as shown in Figure 5. Continuous study is the basis of continuous improvement [9].

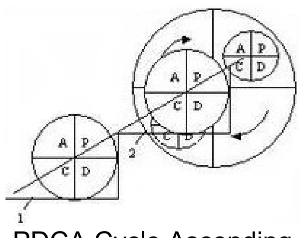


Figure 5. PDCA Cycle Ascending Diagram
1) Original level, 2) New level

As the basic method of quality management, PDCA cycle is not only suitable for the whole software engineering, but also for the whole software enterprises and each department and even individual in the software enterprises. Each department has its own PDCA cycle according to the policy aim of the software enterprise, cycling layer upon layer, in the form of big ring linking with small ring, and small ring linking with smaller ring. Big ring is the matrix and

basis of the small ring, while small ring is the decomposition and guarantee of big ring. PDCA cycle is like climbing stairs, when finishing one cycle, the quality of the production will improve one step, then setting up the next cycle, rerunning and reimproving, thus going forward and improving continuously, as shown in Figure 5. Continuous study is the basis of continuous improvement [9].

## 4. Defects Removal Effectiveness Index

Operation definition of defects removal effectiveness. Definition needs all defect data (including field defect) in the aspects of defect source and in which phase to find and remove defect.

Make $j=1,2,\ldots k$, record the phases of the software life cycle. Make $i=1,2,\ldots k$, record review or test types of different software life cycle phases including maintenance phase (phase $k$). Then the following matrix Figure is defect source/finding place matrix. In this matrix, only Unit $N_{ij}$ (where $i \geqq j$, i.e., the units in the lower left triangle) has data. The data in the units above the diagonal indicate the number of defects injected and detected in the same phase; the data in the units under the diagonal indicate the number of defects injected in the early phase of development but detected in the later phase. For in the early phase, it is impossible to detect the defects injected in the later phase, the units above the diagonal are empty. The boundary row of the matrix ($N_{i.}$) indicate the defects removed in this phase, and the boundary column ($N_{.j}$) indicate the number of defects using this phase as the source, as shown in Table 1.

Table 1. Defects Source/Finding Place Matrix Table

| | Defects Source | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | j=1 | j=2 | j=3 | * | * | * | * | j=k | | |
| i= | $N_{11}$ | | | | | | | | | $N_{1.}$ |
| | * | $N_{22}$ | | | | | | | | $N_{2.}$ |
| | * | * | $N_{33}$ | | | | | | | $N_{3.}$ |
| | * | * | * | * | | | | | | . |
| | $N_{ij}$ (i=j) | * | * | * | * | | | | | $N_{i.}$ |
| | * | * | * | * | * | $N_{ij}$ (i=j) | | | | . |
| Finding Place | * | * | * | * | * | * | * | | | . |
| | * | * | * | * | * | * | * | $N_{kk}$ | | $N_{k.}$ |
| | | | | | | | | | | |
| | N*1 | N*2 | N*3 | * | * | N*j | | N*k | | N Total |

Phase defects removal effectiveness (PDREi) can be phase inspection effectiveness [IE(i)] or phase test effectiveness [TE(i)]: $PDRE_i = \dfrac{N_{i\cdot}}{\sum_{m=1}^{i} N_{\cdot m} - \sum_{m=1}^{i-1} N_{m\cdot}}$

Phase Defects Containment Effectiveness ($PDCE_i$): $PDCE_i = \dfrac{N_{ii}}{N_{\cdot i}}$

Overall Test Effectiveness ($TE$): $TE = \dfrac{\sum_{i=I+1}^{k-1} N_{i.}}{\sum_{i=I+1}^{k} N_{i.}}$ in the formula $I+1, I+2, \ldots k-1$---test phase.

Overall defects removal effectiveness in the development process ($DRE$): $DRE = \dfrac{\sum_{i=1}^{k-1} N_{i.}}{N}$

## 5. Application of PDCA and Rayleigh in Solving Software Quality Management

The target of software quality management is to reduce the height of the curves as much as possible, meanwhile move the crest of the Rayleigh curves to the left, which transform the problem into two indexes, i.e., reducing the injection number of defects at each phase, and increasing the defects phase removal effectiveness. We use PDCA model to solve the problems of reducing these two indexes.

We use the just finished Project A as history data, the scale of Project A is 20000 lines of source code; meanwhile in the new started Project B (the scale is 21000 lines of source code), we use Rayleigh model and PDCA model to make project management. Because the scales of Project A and Project B are basically the same, the work load estimated by COCOMO medium model is basically the same, we first make the following summary on the defects of Project A, as shown in Table 2:

Table 2. Project A Defects Source and Finding Place Example Data Sheet

| Finding Place \ Defects Source | RQ | I0 | I1 | I2 | UT | CT | ST | FIELD | TOTAL |
|---|---|---|---|---|---|---|---|---|---|
| RQ | -- | | | | | | | | . |
| I0 | 20 | 96 | | | | | | | 116 |
| I1 | 37 | 84 | 168 | | | | | | 289 |
| I2 | 83 | 71 | 105 | 339 | | | | | 598 |
| UT | 17 | 40 | 77 | 206 | 37 | | | | 377 |
| CT | 12 | 25 | 72 | 116 | -- | 31 | | | 256 |
| ST | 9 | 31 | 29 | 58 | -- | -- | 17 | | 144 |
| FIELD | 4 | 5 | 13 | 8 | -- | -- | -- | 6 | 36 |
| TOTAL | 182 | 352 | 464 | 727 | 37 | 31 | 17 | 6 | 1816 |

In accordance with the Table, we figure out the defects inspection effectiveness at different phases: IE(I0)=116/(182+352)=21.7%;IE(I1)=289/(182+352-116+464)=32.8%;

IE(I2)=598/(182+352+464-116-289+727)= 45.3%;TE(UT)=377/(182+352+464+727-116-289-598+37)=49.7%; TE(CT)=256/(182+352+464+727+37-116-289-598-377+31)=62%; TE(ST)=144/174=83%; then on the basis of the data, when developing Project B, we use the original development team; before developing Project B, we make relevant summary on Project A using PDCA model, meanwhile apply the summarized experience into Project B, and use PDCA model and Rayleigh model at the beginning phase of Project to make project management to reduce number of defects.

From the above Table, we can see there are many defects injected in different phases, meanwhile the defects inspection effectiveness is relatively low in the phases of high level design, low level design, and coding, etc. Through reason finding, we find the reasons for defects in the high level design are the following defects injection reasons: using wrong parameters, invalid or incorrect screen flow, missing or incorrect of high level flow of components passed in the review package, no input of the module interfaces, incorrect use of the public data structure, unrealized low level design for the code, incorrect variable initialization, etc. We can also see from the above Figure, more defects are introduced in phases such as unit test, components test and system test, which shows that the developers' quality of modifying defects have problems.

Aiming at the above factors for defects injection, the project reviewers make summary, and make plan to prevent the above problems. Ask the experts experienced in design to train the reviewers, be strict in review process; set up defects database, analyze the reasons for these defects, summarize experience, set up check list, check the design contents one by one in the form of check list and cross review method to prevent the similar mistakes from happening again; before modifying defects, the developers first read defects database, meanwhile improve the developers' self test work after modifying defects, to prevent the defects that have occurred from happening again. As per the plan, after we carry out Project B, we get data as shown in Table 3.

From the above Figure, we can see the overall number of defects reduced by nearly a third, in particular, the number of defects in high level design and coding phases reduce obviously. In Project B, the peak value of defects finding is in the inspection phase of low level design, due to the obvious increase of defects removal rate, the peak value also changes, when the peak value reduces, it changes from coding phase to the low level design. The field defects number is also less than that of Project A. So it achieves our targets in software quality management.

Table 3. Project B Defects Source and Finding Place Example Data Sheet

| Finding Place \ Defects Source | RQ | I0 | I1 | I2 | UT | CT | ST | FIELD | TOTAL |
|---|---|---|---|---|---|---|---|---|---|
| RQ | -- | | | | | | | | |
| I0 | 38 | 82 | | | | | | | 120 |
| I1 | 34 | 101 | 254 | | | | | | 389 |
| I2 | 27 | 32 | 96 | 137 | | | | | 292 |
| UT | 29 | 27 | 61 | 51 | 18 | | | | 186 |
| CT | 16 | 14 | 21 | 39 | -- | 12 | | | 102 |
| ST | 9 | 14 | 16 | 21 | -- | -- | 8 | | 68 |
| FIELD | 2 | 4 | 3 | 7 | -- | -- | -- | 2 | 18 |
| TOTAL | 155 | 274 | 451 | 255 | 18 | 12 | 8 | 2 | 1175 |

IE(I0)=28%; IE(I1)=51.2%; IE(I2)=46.6%; TE(UT)=52.8%; TE(CT)=57%; TE(ST)=81%;

## 6. Conclusion

As a reliability model, Rayleigh model applies to the defects analysis in the whole software development cycle, and its remained defects quantitative evaluation mainly relies on the correctness of the early phase data; in qualitative analysis of cross-phase testing activities, single-phase testing evaluation cannot be made. Its significance lies in emphasizing the two principles of defects prevention and early defects removal. They are the main directions of improvement strategy of development quality. PDCA cycle-based software quality management process control and improvement model use process-oriented project plan method, transform the standard process of software organization into the tasks of relevant personnel of software project, which effectively ensure the execution of the quality management process. The analysis of the measurement data enables the objective decision on how to control and improve the quality management process. Combine Rayleigh model with PDCA model in software quality management, and use PDCA cycle in bi-directional quality improvement strategy of Rayleigh model, thus achieving our quality management goal better.

## Acknowledgements

Thanks for the support of the Provincial Natural Scientific Fund Project: The study of the Key Technology of the Web Software Development Platform Oriented to the Service ( No. 20101525) and The Project of Ji Lin Provincial Department of Education: The Development Platform based on OSGi (No. 201085).

## References

[1] Yinglan Fang, Bing Han, Weifu Zhou. Research and Analysis of CMMI Process Improvement Based on SQCS System. *TELKOMNIKA*. 2012; 10(7).
[2] SJ Sun, J Xiao. A Software Reliability GEP Model Based on Usage Profile. *TELKOMNIKA*. 2012; 10(7): 1756-1764.
[3] Xie M. Software Reliability Modeling. Singapore: Word Scientific Publishing Co.Pte.Ltd., 1991.
[4] Lyu M. Handbook of Reliability Engineering.Singapore: McGraw-Hill 1996 covering how to ensure software reliability. *RCA Engineer*. 1982; 27(1): 53-57.
[5] Paulk, Mark C. et al. The Capability Maturity Model: Guidelines for Improving the Software Process (Reading: Addison Wesley Longman, Inc., 1995). Addison-Wesley Publishing Company, Inc. Reprinted by permission of Addison-Wesley Longman. 1995: 81.
[6] DERN INGW E. Out of the Crisis. Cambridge: M IT Center for Advanced Engineering Study. 1982.
[7] Wang Q, Li MS, Liu X. An active measurement model for software process control and improvement. Journal of Software. 2005; 16(3): 407−418. http://www.jos.org.cn/1000-9825/16/407.htm (in Chinese)
[8] Huang Feixue, Li Zhihao, Sun Xiaoli. Study on Quality Guarantee Model Based on PDCA Indian software. J*ournal of Harbin Institute of Technology*. 2005; 11: 1583-1585. Gingrich, Newt Renewing American Civilization, Pillar Five: Quality as defined by Deming. In: Quality Progress (December,1995).