

SOFTWARE QUALITY CONCERN FOR PEOPLE

Proceedings of the
Fourth European Conference on Software Quality
October 17 - 20, 1994, Basel, Switzerland

Edited by



Swiss Association
for the Promotion of Quality
(SAQ)



European Organization
for Quality - Software Committee
(EOQ-SC)

v/dlf

Hochschulverlag AG an der ETH Zürich

Quantitative design of a development process

Zenya Koono and Behrouz Homayoun Far
Department of Information and Computer Sciences,
Faculty of Engineering, Saitama University

Summary:

This paper gives a quantitative design of a software development process aiming at a target error rate. Software errors are built-in by human errors during design at a small rate, and various checks and tests are required to check-out them. Checks and tests also err by the human error rate. Quantitative relations between them have been obtained and explained. The way how to plan error rates at each work process based on past record is shown and the development of each countermeasure is discussed. Quantitative metrics and rational planning as well as accumulating experiences scientifically makes it possible to mature the software work process.

*Saitama University, 255 Shimo-okubo, Urawa, Saitama 338, Japan
phone +81 - 48 - 857 - 4529, fax +48 - 858 - 3716.*

A3

1. Introduction

Well experienced software development leaders forecast the future of a new development of their team very well, predict their individual subordinate's error rates accurately, and manage developments very well based on their forecast and prediction. From their view point, team's development work is like a transmission function which responds to an input generating the corresponding output. It is generally accepted that a development process, consisting of people, the team's way of doing, tools, and their environment, has a repeatable nature. Let's call it a work process. It is a purpose of this paper to reveal quantitative aspect of a work process and to clarify a systematic approach for arranging work process.

In Japanese software industry, Total Quality Control (TQC) has been contributing to improve the quality. The main point of TQC is to improve the work process based on past experiences. Among many practical improvements brought by TQC, a high percentage of improvements are on documents. The effectiveness of documents in development in general, however, has been widely accepted empirically and resulted in ISO 9000. The exact reason why and how documents affect the quality has not yet been clear quantitatively. It is another purpose of this paper to extract systematic aspect way to design work process, and to make the effect of documents clear quantitatively.

Software errors are caused by human errors (Kanno, 1979). Figure 1.1 (Koono, 1987, 1992) shows traces of a software design. A conceptual 'clock' is detailed to a data flow from 'real time clock' to the 'time display'. Successive hierarchical conversions are made to detail the data flow. These data flows are converted to flowcharts and then to source codes. During these mental operations, designers err at a small rate similarly to other human operations, and an error propagates through finally to code. (Koono, 1989, 1993) In Human Reliability Engineering (HRE), this error rate is called Human Error Probability (H_{EP}). (Hayashi, 1984),

(Swain, 1983) H_{EP} shows a wide variation generally. The largest is around 3 times, while the smallest is around 1/3 times of the averaged value respectively (Swain, 1983).

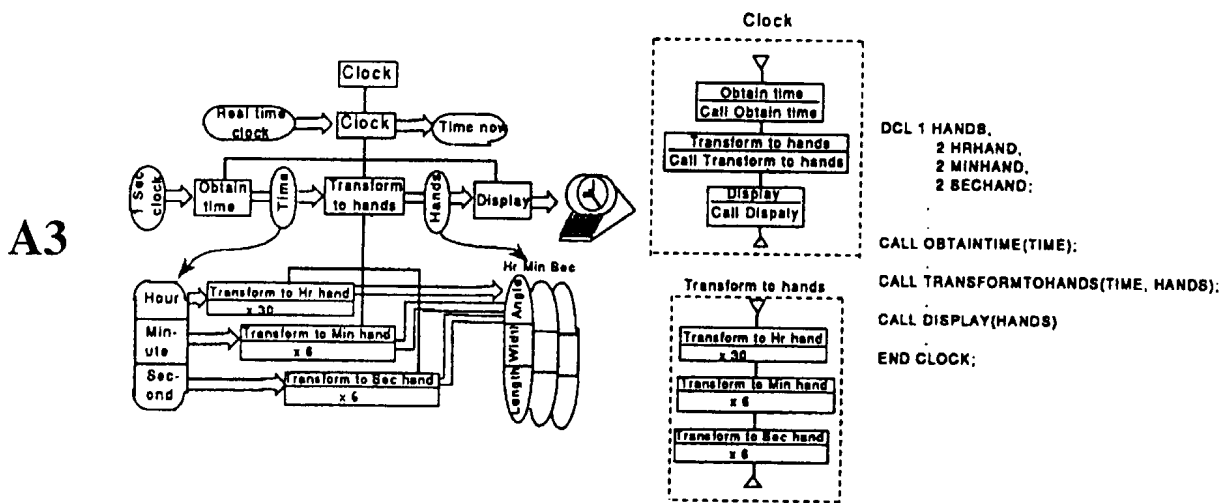


Figure 1.1: Design of 'clock' (Koono, 1987, 1992)

Let us assume that S mental operations are made during designing a software of 1 kilo-lines of non-commentary code (KLOC). The "built-in error rate" is given by:

$$\text{"built-in error rate"} = S \times H_{EP} . \quad \text{Eq. 2.1}$$

Various (desk) checks after design are also human error prone. An example of desk check is to repeat the original design and compare the results. Errors remain when both operations err. Provided that errors occur at random, the "residual error rate" is given by:

$$\text{"residual error rate"} = (S \times H_{EP}) \times (S_c \times H_{EP_c}), \quad \text{Eq. 2.2}$$

where S_c is the number of mental operations during check and H_{EP_c} is the human error probability of the check. In Japanese software industry, "check-out ratio" is often used, which is defined by:

$$\begin{aligned} \text{"check-out ratio"} &= (\text{checked-out errors}) / \text{total errors} \\ &= \{1 - (S_c \times H_{EP_c})\}. \end{aligned} \quad \text{Eq. 2.3}$$

Testing (by machine) is a comparison of the run output of a program by a test input data and the correct output data derived from the specification. It is essentially a comparison of design and test design, both of which are human error prone. The residual error rate after a test is given by:

$$\text{"residual error rate after test"} = (S \times H_{EP}) \times (S_c \times H_{EP_c}) \times (S_t \times H_{EP_t}), \quad \text{Eq. 2.4}$$

where suffix t denotes testing. This shows that test is a kind of check suffering also from errors. (It should be noted that a development is already N-Version program! A complicated test design like Cause and Effect Diagram method might degrade the error rate.)

The typical build-in error rate ($S \times H_{EP}$) per KLOC, is 10-40 errors/KLOC. Usually observed check-out rate is 70-90% or ($S_c \times H_{EP_c}$) is 0.3 to 0.1. Usually ($S_t \times H_{EP_t}$) as a whole is 0.1 to 0.3. (An example of ($S_t \times H_{EP_t}$) is 40 errors/(1000 test cases).) With these values, the final residual error rate of less than 1 error/KLOC had been achieved. Reinforcing desk checks or providing Quality Assurance's test further, the error rate less than 0.1 to 0.01 error/KLOC has been a realistic target.

3. Quantitative planning of error rate

3.1 Build-in error rate (Koono, 1988, 1989)

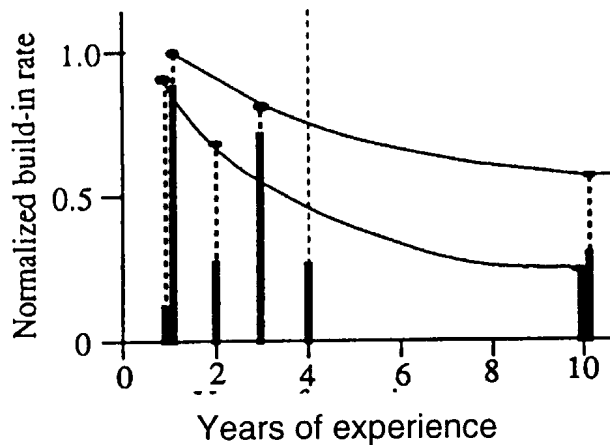


Figure 3.1: Error rate (Koono, 1988)

The usually observed error rate during design is affected by many factors. Figure 3.1 (Koono, 1988) shows an example. It is taken from an initial development of an electronic switching system using assembler language years ago. In this figure, o shows error built-in rate, whereas a black bar shows the apparent error rate, the error rate observed during testing as is popularly used. Plots of o's show a trend curve, while black bars don't. This shows that build-in error rates are intrinsic. (The usual practice is to declare the end of a design just after draft of design documents are finished with rough

A3

check is made. Since then, all errors are recorded or reported, and the accumulated number of errors reported is regarded as the built-in value.)

Figure 3.2 (Koono, 1989) shows various profiles of built-in errors. Samples are taken from various developments using assemblers and compilers. As each project uses individual work process sectioning, work processes near high level design and those near coding level are taken. Around 1/2 errors is built-in during near coding, 1/10 is built-in near high level design and the rest in between them. Man-hour allocations for each design stage were similar each other, and they show a similar trend. When it was changed, it results in a different profile.

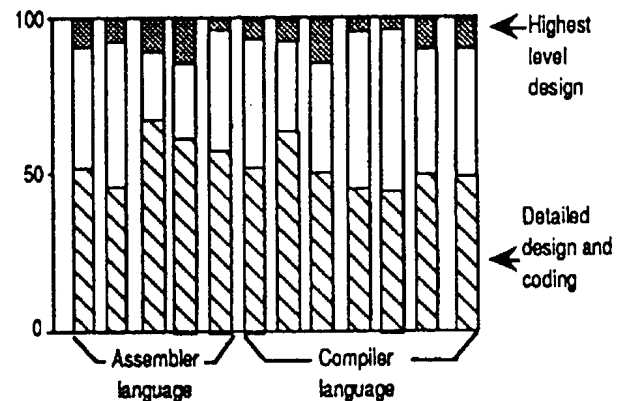


Figure 3.2: Profile of built-in errors (Koono, 1989)

3.2 Check out of errors (Koono, 1988, 1993)

Similar to the stability of "build-in error rate", "check-out ratio" or $\{1 - (S_c \times H_{EPc})\}$ is also stable provided that the work process is kept unchanged. The effect of work process structure is considered here.

- The residual error rate after one time check is $(S \times H_{EP}) \times (S_c \times H_{EPc})$.
- If check is repeated C times, and the error is random, the residual error rate is given by; "residual error rate" = $(S \times H_{EP}) \times (S_c \times H_{EPc})^C$. Eq. 3.1

Repeated checks reduce the residual error rate drastically.

- If design work process is divided to M sections of an equal error rate, and one time check at each section is made, the residual error rates is expressed by;

$$\begin{aligned} \text{"residual error rate"} &= M \times \{(S/M) \times H_{EP}\} \times \{(Sc/M) \times H_{EPc}\} \\ &= \{(S \times H_{EP}) \times (Sc \times H_{EPc})\}/M. \end{aligned} \quad \text{Eq. 3.2}$$

In them, it is important that design result is recorded to be checked later. If it is not recorded, the comparison becomes ambiguous resulting in an increased error rate. But this does not mean that every design needs a particular type of document. Some design documents may be super scribed, and may be checked at each superscription.

A3

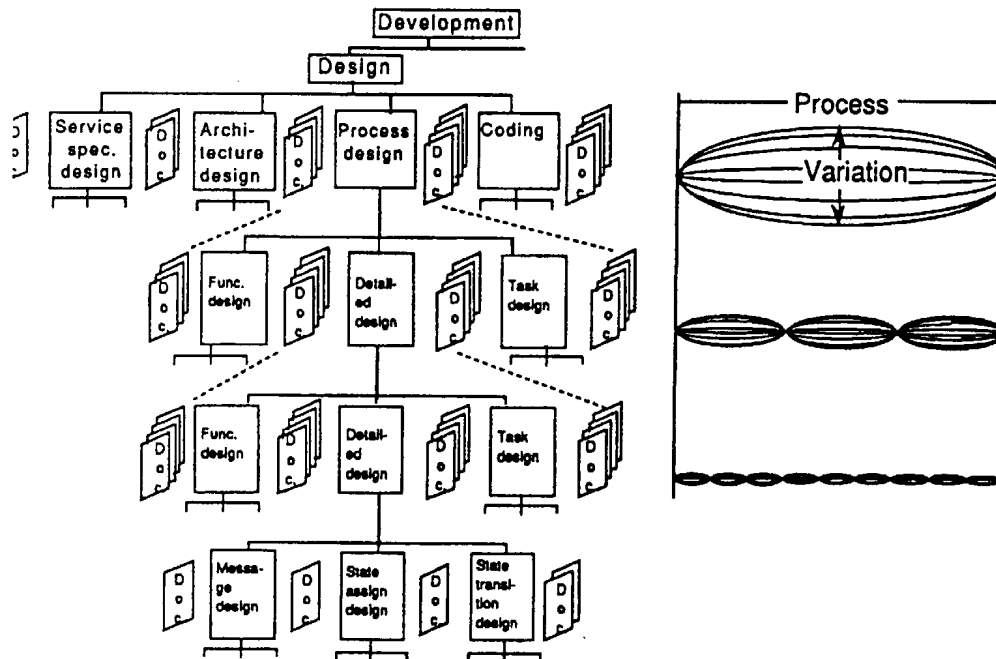


Figure 3.3: Document structure and decrease of the variation

These shows clearly how documents, and therefore work process, should be organized.

- Design records must be left at a small progress of design.
- Documents must be prepared to keep traces of the progress of design.
- Work process is intersected by documents in a higher level or design record blocks which is part of a document.

Figure 3.3 shows a rational document structure fitted for this purpose. A development work process is divided hierarchically to many sections by documents, each document consists of several sub-sections of a work process, and a figure which is a part of a document corresponds to several work procedures during a sub-section. As design progresses, traces of design must be recorded on such documents, and it is very important that they must be checked carefully, rigorously and repeatedly at the end of each procedure, a sub-section and a section.

4. Testing

Testing decreases final residual errors as shown by Eq. 2.4. The error rate before testing is attenuated by a constant. Figure 4.1 (Koono, 1993) is an example. The errors before and after

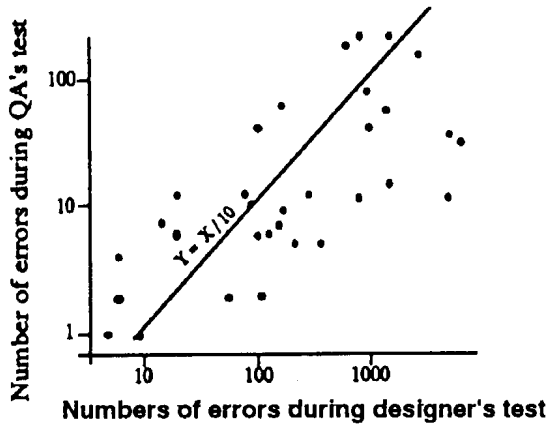


Figure 4.1: Designers' test vs. QA's test (Koono, 1993)

the total attenuation is 1/100. (Sumioka, 92) reports that system simulation testing after them attenuates to 1/4 further.

Figure 4.2 (Koono, 1987) shows another relation on the attenuation of error rates. The horizontal axis shows the accumulated test intensity (accumulated number of test case/KLOC), and the vertical axis is the normalized error rate in logarithmic scale. Plots on the vertical axis is the residual error rate at the beginning of testing. A circle at the end of each graph is the residual error rate at the time of the ship out, which is the number of errors after the ship-out (until they saturate) divided by the software size. The number of errors found in the field is added to the number of errors found during test. A plot on a curve shows the residual error rate after tests of the accumulated test intensity. A broken line shows testing by an environmental simulation test where the number of test is uncountable.

Plots show trend lines, which means that the attenuation rate/test is the same in each development. The gradient of graphs shows the effectiveness of tests, which is a reflection of their work process.

From the figure, effectiveness of test may be obtained:

$$(Et) = (Bt/Bs)^{1/T} \tag{Eq. 4.1}$$

where Et effectiveness of test,
Bs the residual error rate at the beginning of testing,

testing shows a correlation corresponding to an attenuation of around 1/10. (Thayer reported similar data in (Thayer, 1976).) Author's colleague Dr. Watanabe found that the variation is decreased when groups sharing the same work process are taken.(Watanabe, 1982) Authors and Dr. Watanabe's data show that both designers' (rather white box) tests vs. Quality Assurance(QA)'s (including black box) tests, and QA's tests vs. field errors show approximately 1/10 attenuation, therefore

A3

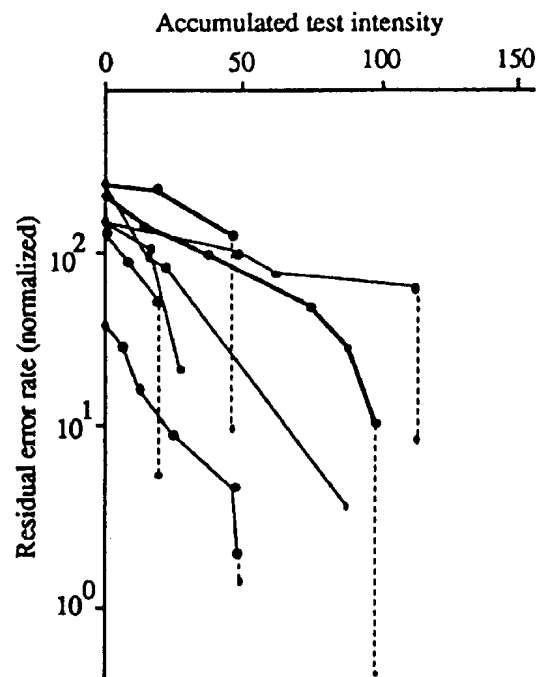


Figure 4.2 Attenuation of error rate (Koono, 1987)

When the work process is kept, the effectiveness of test is also kept. It enables the quantitative design of test possible. The effectiveness of tests in the figure are in two groups. The low effectiveness group is bad work process one where documents were poor, and team's morale is low. They feature very high cancellation rate of design changes issued previously by bugs. These cancels of high rate were caused by unclear or insufficient information in documents. The situation is shown by Figure 4.3. The discrepancy of design information from what it

A3

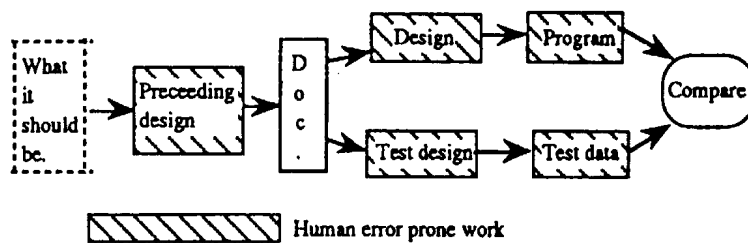


Figure 4.3: Structure of test (Koono, 1990)

should be brought this problem. Documents are essential also for testing. (Errors during testing has been neglected, though they are apparent when one tests by oneself. Quantitative evaluation on them leads to various evaluation of design methods.)

5. Quantitative planning of work processes

"Build-in error rate" during design, "check-out ratio" by desk checks and the effectiveness of test have been described. These are essential and intrinsic metrics of quality of a software development work process. In order to improve the quality, it is necessary to improve the work process. From present metrics, an error level diagram shown in Figure 5.1 is obtained.

- 1) The improved status may be designed quantitatively as shown also in Figure 5.1. The first step is to fix the target ship out error rate, then the improvement is partitioned to design and test. (In this figure, improvements on check-outs by desk checks and tests are considered to give a safety side evaluation.)
- 2) The decrease during test is reduced from $5.5/44 (= 1/8)$ to $1/22 (= 1/22)$. Assuming that the effectiveness of test is kept constant, the required increase of the test intensity is given by

$$\text{Increase ratio of test} = \text{Log}(1/22)/\text{Log}(5.5/44) = 1.49.$$

In order to attain this, it is better to introduce Quality Assurance test of the half number of the previous test intensity.

- 3) Based on the present check-out rates, each desk check system, and the present document system, the improvement may be designed. If check-out rates are not available in each small work process, a simplified version as shown in Figure 5.1 may be used for the initial phase. For a moderate increase of check-out, improvements on desk check system is adequate. For substantial increase of check-out rate (e.g. 0.5 to 0.7) or substantial decrease of $(Sc \times H_{EPc})$ (e.g. 0.2 to 0.05), sectioning to smaller work processes with checks at each subdivided section is taken. If necessary, new documents is introduced. If document system is enough, the work process to complete a document is divided to several sections and checks at each section is reinforced. The principle is shown by Figure 3.3. In this case, process design is equally divided to 3 parts, the residual error rate decreases to $1/3$. When it is not sufficient, it is divided further as shown.
- 4) People requires more man-hours during design to increase documentation and checks. Man hour needed for the increased documentation must be added. (However, it will be usually compensated by the decrease of debugging man-hour during testing.) As people

(the most influential component of a work process) can not be changed abruptly, a moderate advanced plan by rational management is necessary. Various means to enable the target check-out rate must be provided. Various strategies and tactics are well described in Total Quality Control text books.

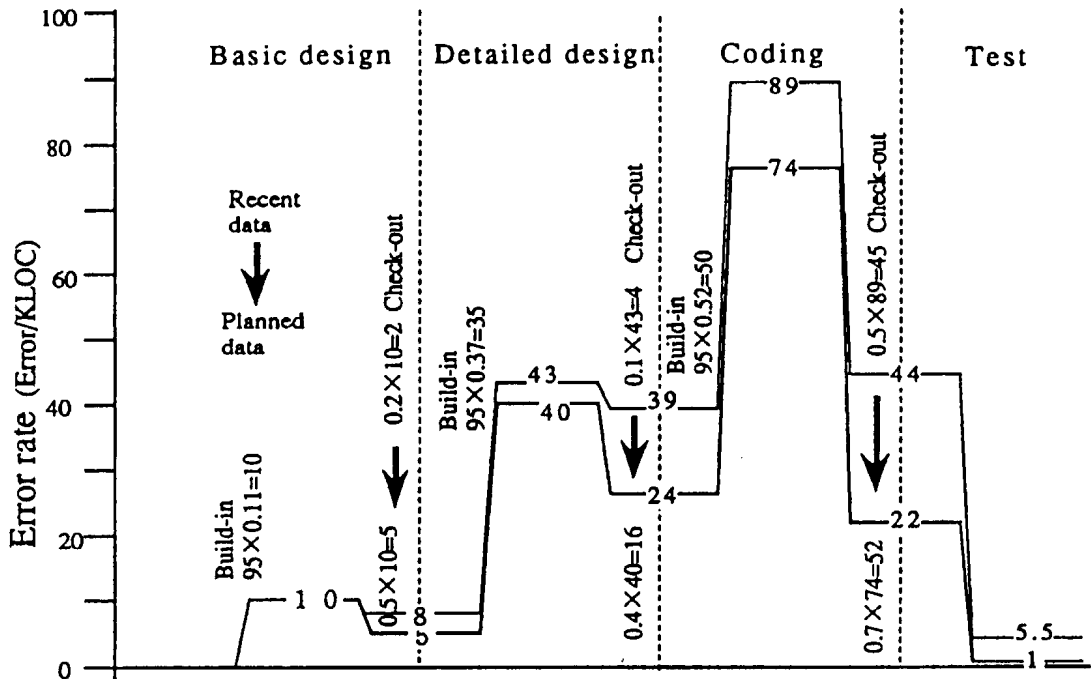


Figure 5.1: Error level diagram

- In the new project, build-in and check-out rates must be measured in revised small sections. After the development, various data should be evaluated. Build-in and check-out rates of more divided to small sections are available. More accurate understanding of work process becomes possible. (Build-in rates are usually decreased.) Based on them, work process improvements becomes possible for leaders, and improvements not to repeat the same kind of errors may be attained by their root cause analysis.

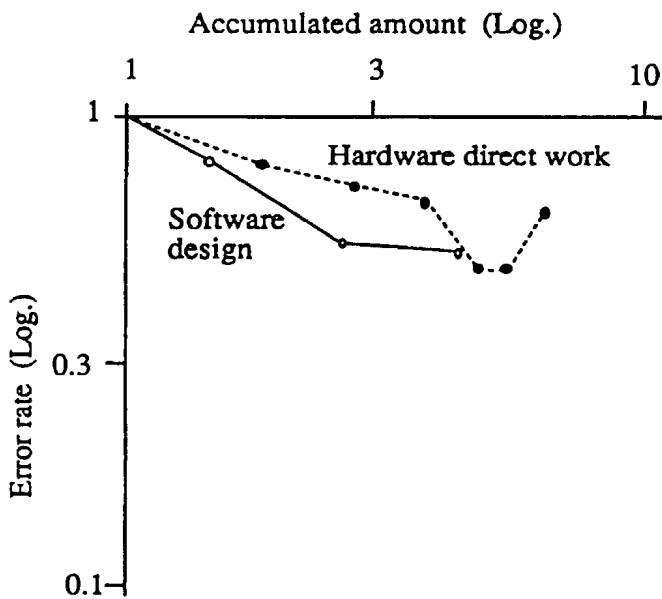


Figure 5.2: Learning curve of error build-in rate (Koono, 1988)

By repeating these Plan-Do-Check-Action (PDCA) cycles, the error rate of a team is improved.

The progress shows a maturity as pointed out. The intrinsic metrics follow a Learning effect as

shown in Figure 5.2.(Koono, 1988b, 1989, 1990) Thus accumulating improvements by quantitatively measurements, the work process advances. A high level of maturity may be attained thus continuing rational improvements for a long period. It is an advance from primary school level to graduate school level.

6. Conclusion

A3

A quantitative design method of software error rate has been reported. Software errors arise from inevitable human errors not from the complexity of the problem. Assuming Human Error Probability, a systematic model for error build-in and check-out of errors (by desk check and testing) was proposed. This also makes it possible to evaluate work processes of intersections by documents and effectiveness of checks and tests. Using this modelling, it becomes possible to design a software error rate at each stage.

When enough knowledge exists, the work processes for attaining the planned error rates at each stage may be determined.

This quantitative ways reported in this paper are based on scientific and rational structures of developments, which are implicitly used by many experts in an incremental basis. This is just a justification of what experts do. The most important thing in engineering lays in the forefront of the work.

Acknowledgements

Various rules and relations in this paper are extracted from what experts are doing. The authors wish to express their thanks to many leaders and designers. The true author of this paper are they. The authors also express their appreciation to those who participated in this study. The authors are also grateful to their bosses for enabling this study.

References

- (Hayashi, 1984) Hayashi Y.: Human reliability engineering. Kaibundo, 1984. (in Japanese)
(Japanese title: 林喜男, "人間信頼性工学", 海文堂, 1984.)
- (Kanno, 1979) Kanno, A.: Software Engineering - Software Development, Production and Quality Control. JUST, 1979. (in Japanese)
(Japanese title: 菅野：ソフトウェアエンジニアリング ソフトウェアの開発 - 生産と品質保証, 日科技連, 1979.)
- (Koono, 1987) Koono, Z. and Ashihara, K. and Soga, M.: Structural way of thinking as applied to development. IEEE/IEICE Global Telecommunications Conference 1987, 1987.
- (Koono, 1988a) Koono, Z., Yamato, Y. and Soga, M.: Structural way of thinking as applied to high quality design. IEEE International Conference on Communications 1988, 1988.
- (Koono, 1988b) Koono, Z. Igawa, K. and Soga, M.: Structural way of thinking as applied to improvement process, IEEE Global Telecommunications Conference 1988, 1988.
- (Koono, 1989) Koono, Z.: Structures of quality progress and developments. The 10th Software Reliability Symposium, 1989. (in Japanese)
(Japanese title: 河野：開発作業と品質向上の構造, 第10回ソフトウェア信頼性シンポジウム, 1989.)

- (Koono, 1990) Koono, Z.: A model for building-in and checking-out of errors, The 11th Software reliability symposium, FTS research group, The Institute of Electronics, Information and Communication Engineers, 1990. (in Japanese)
(Japanese title: ソフトウェア不良の作り込みと除去のモデル, 第11回ソフトウェア信頼性シンポジウム, 電子情報通信学会FTS研究専門委員会, 1990.)
It is published in English in Koono, Z. : Build-in and Check-out of software errors, The third International Workshop on Software Quality Improvements '91, 1991.
- (Koono, 1992a) Koono, Z.: Structural way of thinking as applied to increasing customer's satisfaction. The 3rd European Conference on Software Quality, 1992. (Koono, 1992b) Koono, Z. and Far, B. H.: Quantitative Basis for software development process, Technical Report, KBSE-92-33, The Institute of Electronics, Information, Communication Engineers, 1993. (in Japanese)
(Japanese title: ソフトウェア開発工程の定量的取り扱い, 信学技報 KBSE92-33.)
- (Koono, 1993) Koono, Z. and Ohtsubo, T.: Evaluation of build-in and check-out of software errors. Technical paper SE-95-5, The Information Processing Society, 1993. (in Japanese)
(Japanese title: 河野, 大坪: ソフトウェアの誤りと除去の評価, 情報処理学会, ソフトウェア工学研究会資料 95-5, 1993.)
- (Koono, 1994a) Koono, Z. and Far, B. H.: Structure of software development process - From a view point of Industrial Software Engineering, Technical paper SE-98-5, The Information Processing Society, 1994. (in Japanese).
(Japanese title: 河野ファー, : ソフトウェア開発プロセスの構造, 情報処理学会, ソフトウェア工学研究会資料, ソフトウェア工学98-5, 1994.)
- (Koono, 1994b) Koono, Z. and Far, B. H.: Structural way of thinking for attaining reliable software, IEEE International Conference on Communications 1994, 1994.
- (Sumioka, 1992) Sumioka, J., Shimizu, M., Uryuu, S. and Osawa, Y. : System simulation testing: Pre-delivery testing to ensure systems dependability, 3rd European Conference on Software Quality, 1992.
- (Swain, 1983) Swain, A. D. and Guttman.: Handbook of human reliability analysis with emphasis on nuclear power plant, NUREG/CR - 1278, SAND 80 - 0200 (1983)
- (Thayer, 1976) Thayer, T. A. et al.: Software reliability study, TRW System Rep. SS-76-03, 1976.
- (Watanabe, 1982) K. Watanabe and Ogata, H.: A prediction method for software productivity and quality. The 2nd Software Quality Control Symposium in Software Production, 1982. (in Japanese)
(Japanese title: 渡辺, 緒方. : ソフトウェアの品質および生産性予測法の一事例, 第2回ソフトウェア生産における品質管理シンポジウム, 1982.)