Toward Quantitative, Rational and Scientific Software Process

Zenya Koono[1] and Hui Chen[2]

[1] Nara Institute of Science and Technology, Honfujisawa 2-12-5, 251-0875, Japan
koono@vesta.ocn.ne.jp

[2] Center for Information Science, Kokushikan University, 154-8515, Tokyo, Japan
chen@kokushikan.ac.jp

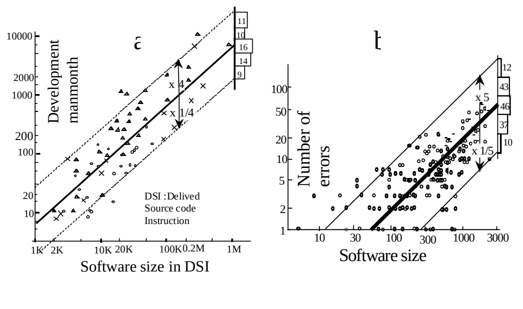
Abstract. The most important characteristics of a software process are those of product, cost and quality. For this purpose, this paper proposes a quantitative model based on the scientific nature of a software process.

1 Introduction

It is well known that quantitative basis of a process is indispensable for the rational and scientific evaluations and improvements. But, that for a software process has not yet been established. The purpose of this paper is to propose a quantitative model[1~4]. Based on field data, a simple mechanism, resulting in the characteristic, is derived. By repeating it, a whole model of a software process is obtained. This measures AS-IS characteristic of a process, independent of ways of the development, and it enables to improve a process toward any direction needed.

2 Fundamental relationships

Fig. 1 shows software size vs. man-hours and errors of actual data[5~6]. The center bold line is the main trend line ($Y = AX^1 + B$) of plots. Following may be obtained:



- S/w process is a linear system, where decomposition as well as integration is possible.
- Man-hours and errors are proportional to s/w size, when the process is kept.
- Plots distribute in a lognormal distribution, ranging from 1/N to N (where N = 4~5) of the center value. (Max/min) ratio ranges from 16 to 25.

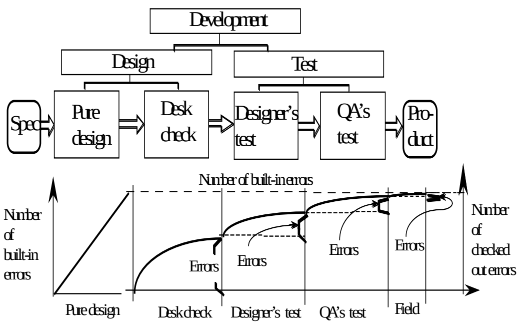


Fig. 2 Error accumulations

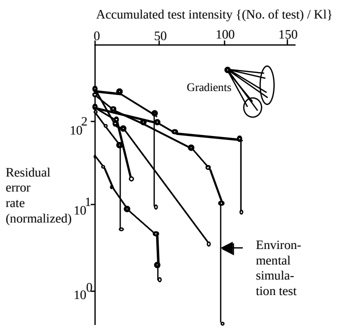


Fig. 3 Decrease of error by test

Figure 2 shows accumulation curves of errors. *Pure design*, in the most left, builds in errors at a rate. In following processes, errors are checked out as shown.

- Built-in number of errors is obtained by summing up all errors found in following processes. Thus gained built-in error rate is stable. (Most studies neglect desk check number. It results in variation of around 0.5 ± 0.3. This invites mystery of error.)
- Curves show (1 - e^{-ax}) shape. It is estimated that both *test* and *desk check* is error attenuators of a certain rate. It will result in a linear trend line in a logarithmic scale.

Figure 3[7] shows various tests in logarithmic scale. The gradient is the decreasing rate of each test, and is named as *the effectiveness of test*.

The most right side plot is (X= total test intensity, Y = error rate found in the field). In the next left plot, X is decreased by the last stage test intensity and Y is added the error rate found there. Finally, the most left side plot is (X =0, Y = residual error rate before test or after desk check).

Summarizing, a software process shows following quantitative characteristics.

1. A process consumes man-hours by its *productivity* (e.g. Man-hour/KLoc).
2. Pure design builds in errors by its *error (build in) rate*.
3. Desk check and test attenuate error rate by respective *error-decreasing rate*.

Figure 4 shows the strategy to stabilize a process. A process is intersected in a hierarchical manner by hierarchical documents. As the hierarchical division goes down, procedures are standardized more finely or constrained more strongly, thus variations decrease as shown in the right side. (It is widely used in h/w production process.)

Desk check should be made at the end of each process. Figure 5 explains the way. If a process is divided to M sections of *pure design* and *desk check*, the residual error rate is attenuated by 1/M, as shown below. (Also this is proved in h/w production.)

In the top figure, design errs at rate of Ed, and desk check does at a rate Ec. Let us assume Ec is the *second class of error of test*, which is the probability of mistaking NG item as Good one. After the check, the resultant error rate is Ed·Ec, namely errors are decreased by Ec. When both are divided to M equal processes, the resultant error rate is given by M x (Ed/M)·(Ec/M) = (Ed·Ec)/M.

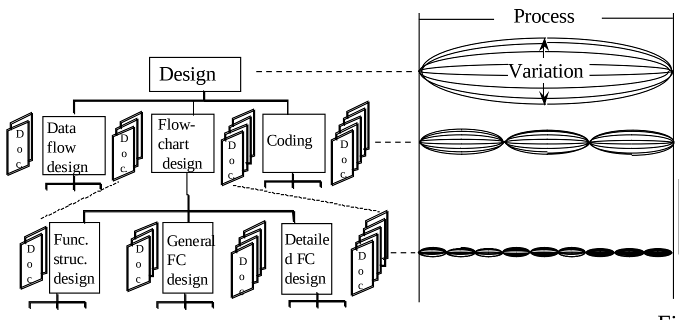


Fig. 4 Constraints and variation

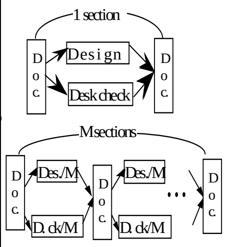


Fig. 5 Ways of design and check

From these, it is understood that so-called heavy process features high quality, low error rate and good stability for meeting industrial needs of reliable production.

Aforementioned characteristics well represent the present status of a process. In order to make them applicable in any case, following expansions are made. Fig. 6 shows man-hours (MHs) of test for 4 sub-systems (teams) of a system's initial development. The intercept on the ordinate is MHs for *pure test* without errors. From it, MHs for a test may be obtained. The gradient of the trend line is MHs for repairing (including re-testing) an error found. From these, MHs for test of any test intensity and any error rate may be obtained by the following equation.

(MHs for a test)·(number of test)+ (MHs for an error)·(number of errors)

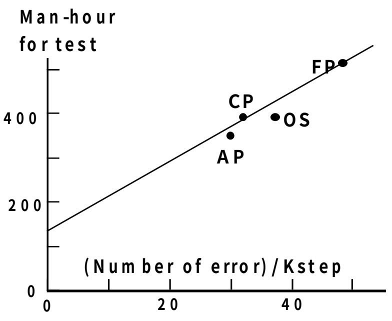


Fig. 6 Test man-hours

MHs for desk check, for meeting the specified residual error rate, may be obtained as follows. Let define C be normalized desk check man-hour as (MH of desk check) divided by (MH of pure design.) During desk check, the normalized residual error rate D is given by $D = e^{-aC}$. Based on an initial experience of C_0 and D_0, the necessary normalized man-hours C_1 for attaining the specified residual error rate of D_1 is given by $C_1 = C_0 \cdot (\mathrm{Ln}\ D_1 / \mathrm{Ln}\ D_0)$. Using these, MHs for a next project may be planned rationally.

Productivity, error building in rate, decay constant of desk check and *effectiveness of test* as well as their variations are intrinsic characteristics specific to a process (e.g. a team). They follow logarithmic learning effect[9]. They may be improved *only when improvements are made on the process by their own efforts*. When improvements are made quantitatively and rationally by all the people, from top managers to engineers, their learning effect curves grow linearly on a both logarithmic chart.

3 Discussions

From aforementioned characteristics, various overall quantitative views of the process may also be shown by integrating them based linear nature of process. Fig. 7 shows data of an excellent development[8] by GTE Lab in late 1970's. The curve in Fig. a shows the accumulated error rate as an error is found. 82.5% of errors are checked out prior to machine test. The bottom bar graph shows break down of man-hours, and it show that they used 30~50% man-hours of *pure design* for *desk check*. Fig. b shows cycles of, build in by *pure design* and check out by *desk check*, for each design phase. Fig. c shows attenuations during machine tests. These graphs show overall status of the project.

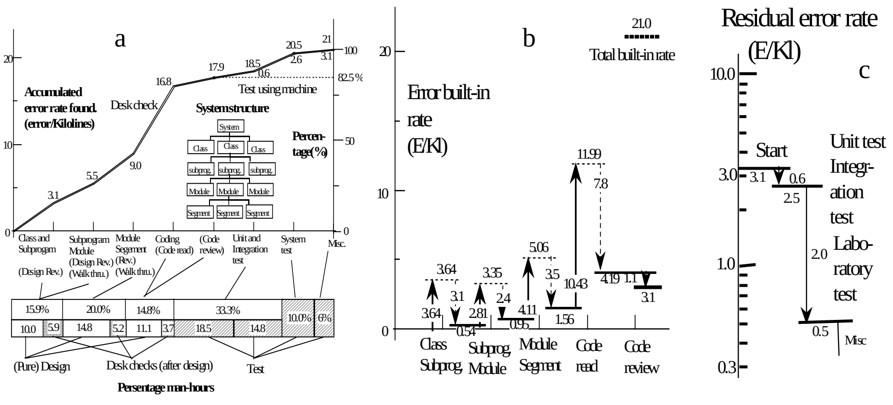


Fig. 7 An excellent development example

Basic standing point is actual characteristics experienced. All characteristics are specific to a process. In order to force same target values to a whole group, a common process in that group has to be prepared at as is done in Japan's software factory[10].

Various characteristics discussed may be applicable also to business process, development of systems as well as hardware. As they are extensions of those used in hardware manufacturing process, they may be used also there. As this model bases on Human Intelligence, common to all kinds of human intentional activities, such wide capability is attained. By this quantitative model, many established techniques, on human processes, become applicable. They are project management (production control in Industrial Engineering (IE), productivity improvements (Time Study and Value Engineering in IE), quality control (QC in IE) and quality improvement (Total Quality Management).

In non-repetitive development area, R&D management is the crucial key for success. This excellent team's ways are worth to be noted. 1. Scientific attitude to new techniques, 2. Quantitative evaluations, and 3. Field trial prior to full deployment. These by excellent managers matured this team to attain such high level.

4 Conclusion

In this paper, a quantitative model, based on simple approximations, has been proposed. As it shows AS-IS, it features simplicity, wide applicability and high usability, and independent of various methods on processes. Using this with 'Divides and conquers' of a process, and repeating improvements, a process may be tailored to any direction. Authors wish to introduce various ways[2] to use it in the next opportunity.

Acknowledgement

One of authors, Koono, wishes to express deep gratitude to peoples in fields of Industrial Engineering, Production, Quality Engineering and software in Totsuka Works, Hitachi, Ltd. They also thank to Dr. B. H. Far and Dr. Hassan Abolhassani as well as students of Saitama University engaged in Software Creation Project. They are also thankful to people in EASE project for their valuable discussions. .

References

1. Koono, Z., Chen, H. and Far, B. H.: Expert's Knowledge Structure Explains Software Engineering. Joint Conference on Knowledge Based Software Engineering (1996) 193-197
2. Koono, Z. and Chen, H.: Structure of Human Design Knowledge and the Quantitative Evaluation (Part 1/2). Technical report of IEICE, KBSE2003-57 (2004) 67-72. (In Japanese)
3. Koono, Z. and Chen, H.: Structure of Human Design Knowledge and The Quantitative Evaluation (Part 2/2). Technical report of IEICE, KBSE2003-57 (2004) 73-78. (In Japanese)
4. Koono, Z. and Chen, H.: A Software Process Featuring Quantitative Model. Technical report of IPSJ, Technical report of IPSJ 2005-SE-147 (12). (In Japanese)
5. Boehm, B. W.: Software engineering economics. Prentice Hall (1981)
6. Thayers, T. A., et al.: Software reliability study. Final Technical Report, RADC-TR-76-238, Rome Air Development Center (1976)
7. Koono, Z., Ashihara, K. and Soga, M.: Structural way of thinking as applied to development. IEEE COMSOC Global Telecommunications Conference (1987) 26.6.1-6
8. Reproduced by authors' responsibility from Daly, E. B. and Mnichowicz, D. A.: The management of large software development for stored program switching systems. International Switching Symposium (1979) 1287-1291 and GTE internal documents
9. Hancock, W. M. and Bayer, F. H.: Learning curves, Salvendy, G. eds. Handbook of Industrial Handbook. John Wiley & Sons (1982) 243-251
10. Cusumano, M. A.: Japan's software factories: A challenge to U.S. management. Oxford Press (1991)