Risk analysis of software process measurements

Tsutomu Kojima*, Toru Hasegawa*, Munechika Misumi** and Tsuyoshi Nakamura***

*Division of Software Process Engineering, Software Research Associates, Inc., Toshima-ku, Tokyo 171-8513, Japan

**Department of Statistics, North Carolina State University, Raleigh, NC 27695-8203, USA

*** Department of Biostatistics, Graduate School of Science and Technology, Nagasaki University, Nagasaki 852-8521, Japan

Abstract

Quantitative process management (QPM) and causal analysis and resolution (CAR) are requirements of capability maturity model (CMM) levels 4 and 5, respectively. They indicate the necessity of process improvement based on objective evidence obtained from statistical analysis of metrics. However, it is difficult to achieve these requirements in practice, and only a few companies have done so successfully. Evidence-based

risk-management methods have been proposed for the control of software processes, but are not fully appreciated, compared to clinical practice in medicine. Furthermore, there is no convincing answer as to why these methods are difficult to incorporate in software processes, despite the fact that they are well established in some business enterprises and industries. In this paper, we challenge this issue, point out a problem peculiar to software processes, and develop a generally applicable method for identifying the risk of failure for a project in its early stages. The proposed method is based on statistical analyses of process measurements collected continuously throughout a project by a risk assessment and tracking system (RATS). Although this method may be directly applicable to only a limited number of process types, the fundamental idea might be useful for a broader range of applications.

Keywords logistic model, risk assessment, software process, statistical analysis, yore, temodori

1.Introduction

The software application field has been expanding dramatically, and the software development process has become more complex, resulting in an ever-increasing demand for reliable software. (De Lacalle et al., 2002; Maydl, 2004; Ingham et al., 2005) The environment in which software development currently takes place is more challenging than conducive to success, and the demand for skilled and experienced managers is increasing (Pfahl et al, 2003; Ellis et al, 2004). Moreover, software development technology changes every few years, thereby limiting the availability of expert managers. Consequently, it is becoming increasingly difficult to develop a product of the required quality within a specified time frame (Kang et al, 2005), which may have serious effects on software manufacturers, vendors, and users. Therefore, how to produce a high-quality system in a timely manner is one of the most critical and important themes of project management.

Discussing this issue, Bieman (2004) argued: “The only hope for making informed design decisions leading to systems that remain high-quality and adaptable is to improve the ability of designers to prognosticate. Rather than use a crystal ball, comprehensive studies of how existing systems have evolved in the past can provide solid evidence into the

connection between early design decisions and the evolving adaptability and quality of software systems. ……We can improve our ability to perform relevant prognostication only with a much deeper understanding of how systems have evolved."

This argument seems relevant to a recent software glitch that troubled the Tokyo Stock Exchange in Japan. In December 2005, a simple input error led a brokerage firm to lose US$400 million in a few hours. (Williams 2005). Those who noticed the error tried to correct it from their terminals, but the system controlling the trade did not accept the correction. It is generally thought that the glitch was partly because the software engineers who developed the system did not foresee the simple input error. Conversely, the software engineers insisted that the product-testing period has been gradually shortening over the past 10 years (Hirayama et al, 2004; Kang et al, 2005), and suggested this as the real cause of the glitch. If this is indeed true, software vendors should be willing to pay more for more sufficient testing.

To address this issue and improve prognostication requires solid evidence about the current situation, and the development of a plan for moving forward. We propose the following method: collect data on various prognostic variables throughout the system

development process, including testing periods, before release; collect follow-up data after release to identify any bugs in the system within a prescribed period of time; and develop a list of critical prognostic variables to distinguish between successful and failed systems by a comprehensive statistical analysis. If the testing period were then detected as statistically significant in the analysis, this would indicate that this factor is indeed partly responsible for product glitches.

1.1. The Risk Assessment and Tracking System

Since 1994, the Software Research Associate (SRA) has started several actions to implement the above idea. In 2004, it combined these strategies into one, called the risk assessment and tracking system (RATS), which aims to identify and track risks to detect and resolve problems in the earlier stages of system development. Figure 1 shows a graphical representation of RATS. Our proposed risk-management strategy, described above, is implemented as follows:

(1) Identify risks based on initial data.

(2) Evaluate initial risks.

(3) Determine what level of management will be in charge of the project.

(4) Track data on progress and quality.

(5) Identify risks based on the tracking data.

(6) Evaluate progressive risks.

Steps (3)–(6) are repeated continuously throughout the life of the project.

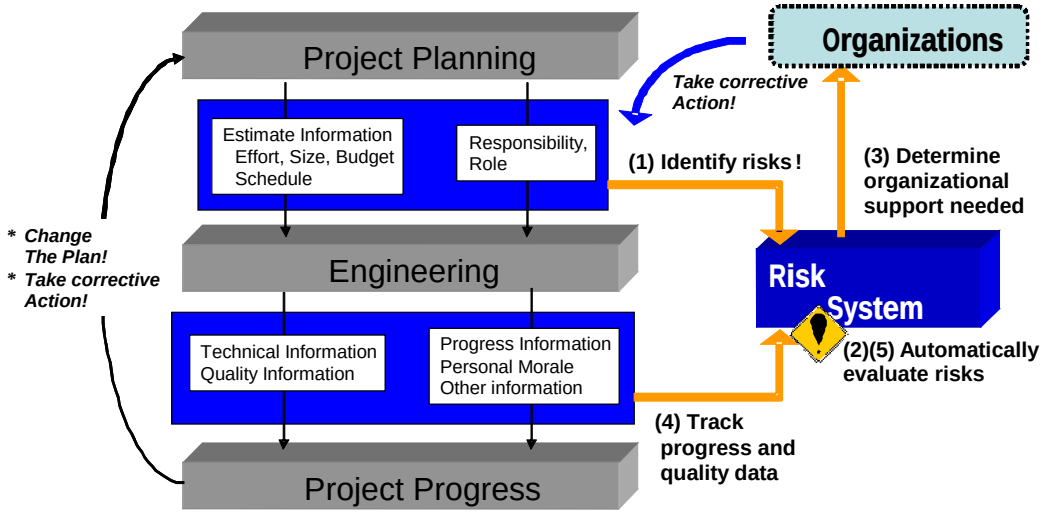

Fig. 1. RATS system for risk management

Processes such as this are not novel, but are usually difficult to establish because they often place a large burden on project members, especially project managers, making implementation impractical. To avoid placing an additional burden on project members, we have linked risk management and progress management. Specifically, our system automatically analyzes and evaluates risk as information about project progress and quality is inputted.

1.2 Risk visualization

Figure 2 shows the results of evaluating risk automatically. Each point in the graph represents the risk level of each project. The horizontal line denotes the field risk of revealing their risk level, calculated using an equation based on an engineering perspective, and the vertical line represents the management risk of revealing their damage level from failure based on an administrative perspective. In RATS, this result is a starting point. Then, changes in the risk level are tracked throughout the project. Consequently, risk visibility, early detection, and resolution become possible, while there is no increase in the burden placed on project management.

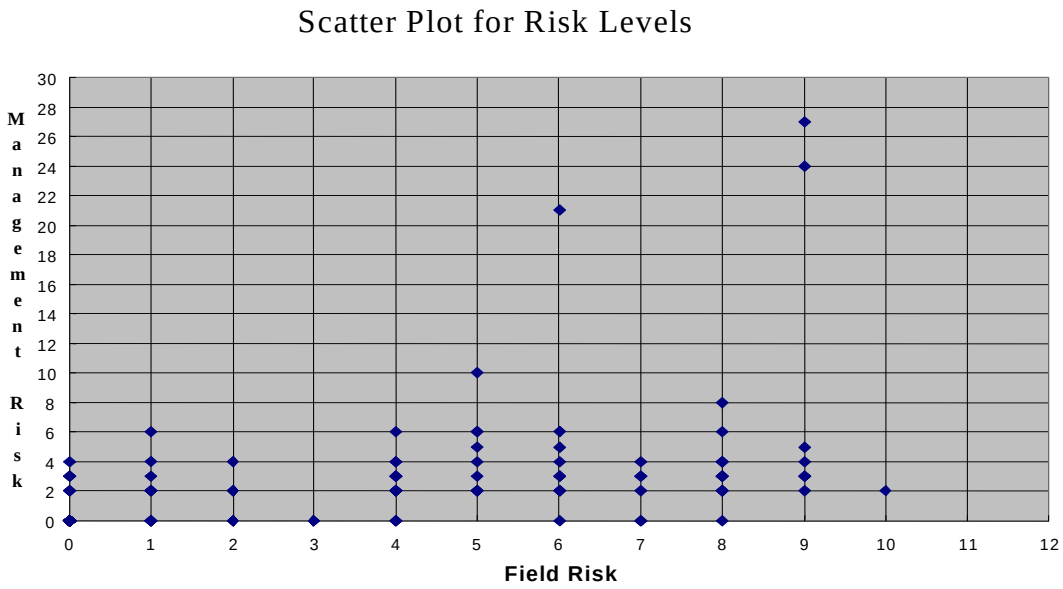

Fig. 2. Scatter plot for field risk vs. management risk to determine the management level in charge of the project

Next, we forecast the project progress based on the accumulated data in RATS. To this end, we apply comprehensive statistical analyses to identify any characteristics specific to the software process that might be preventing the effective use of software metrics for process improvement.

1.3 Challenges from a previous work

Figure 3 shows a scatter plot diagram for the estimated gross profit at the start of a project vs. the actual gross profit at project completion. The graph reveals that there were three big projects whose profits were ultimately far below initial estimates. Identifying such projects heading toward failure at an early stage is the first challenge this study addresses. To do so, a multiple regression model can be used to analyze the relationships among the process measurements within the first month and the estimated and actual gross profits (Kojima et al., 2005). Kojima et al. (2005) also discussed how to correct process measurement estimates to obtain a corrected actual profit. Although the multiple regression method works well for most applications (Khoshgoftaar et al 1994, Liu 2006, Khoshgoftaar et al 2006), it is vulnerable to outliers, or extremely large measurements, and thus is not sufficient for many large projects. The challenges to applying this method, revealed by Kojima et al. (2005), are summarized below:

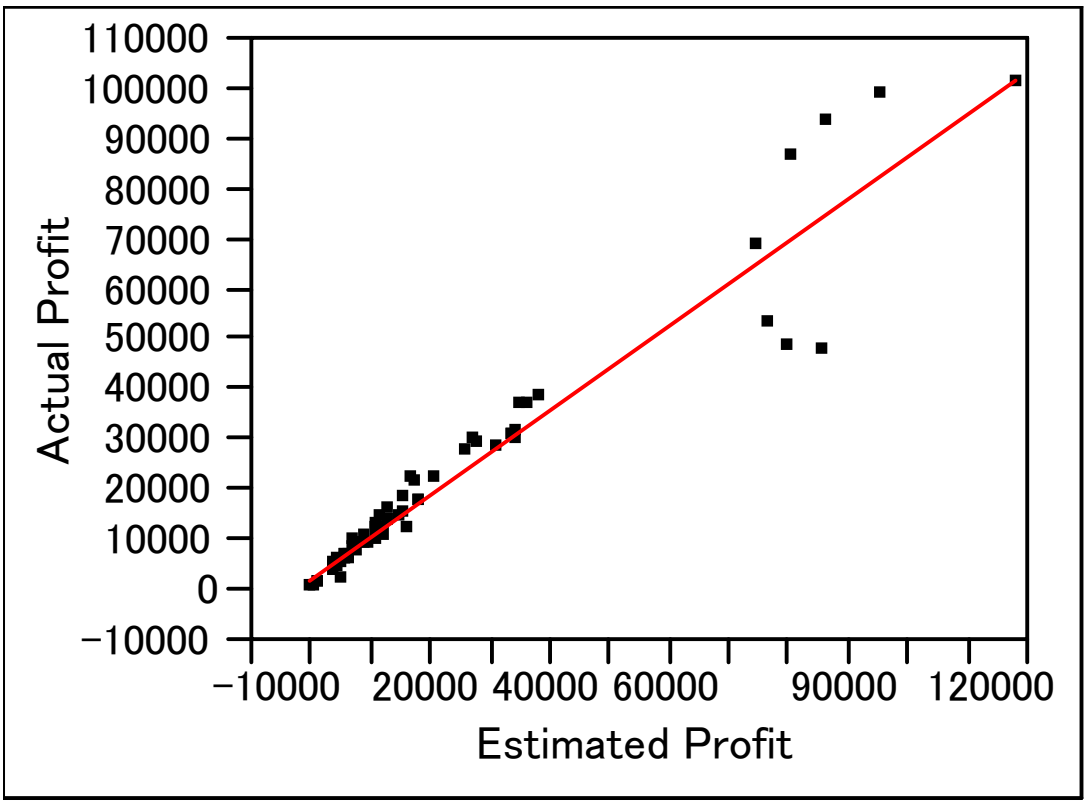

Fig. 3. Scatter plot of the estimated profit at the start of the project vs. the actual profit at the end of the project

(1) To establish a reliable model for predicting failed projects, a considerable number of projects with available variables is required. However, missing data are common, which severely limits the number of projects available for the analysis;

(2) Although some statistical methods, such as the logistic model, Cox proportional hazards model, and multiple regression model, are useful for most practical applications, it remains unclear how and when those methods can be effectively applied to software process improvements;

(3) Extreme values are considered statistical “outliers”, which are usually viewed as obstacles in statistical analyses, but they may in fact be preferable in the case of project prediction, because they could represent large profits. However, it is difficult to accommodate both normal size projects and extremely large projects simultaneously in a multiple regression model;

(4) Although Kojima et al. (2005) successfully analyzed their data using the log-transformation of original measurements, even this method does not always produce reliable results when applied to broader applications, due again to the vulnerability of the multiple regression method to extremely large values.

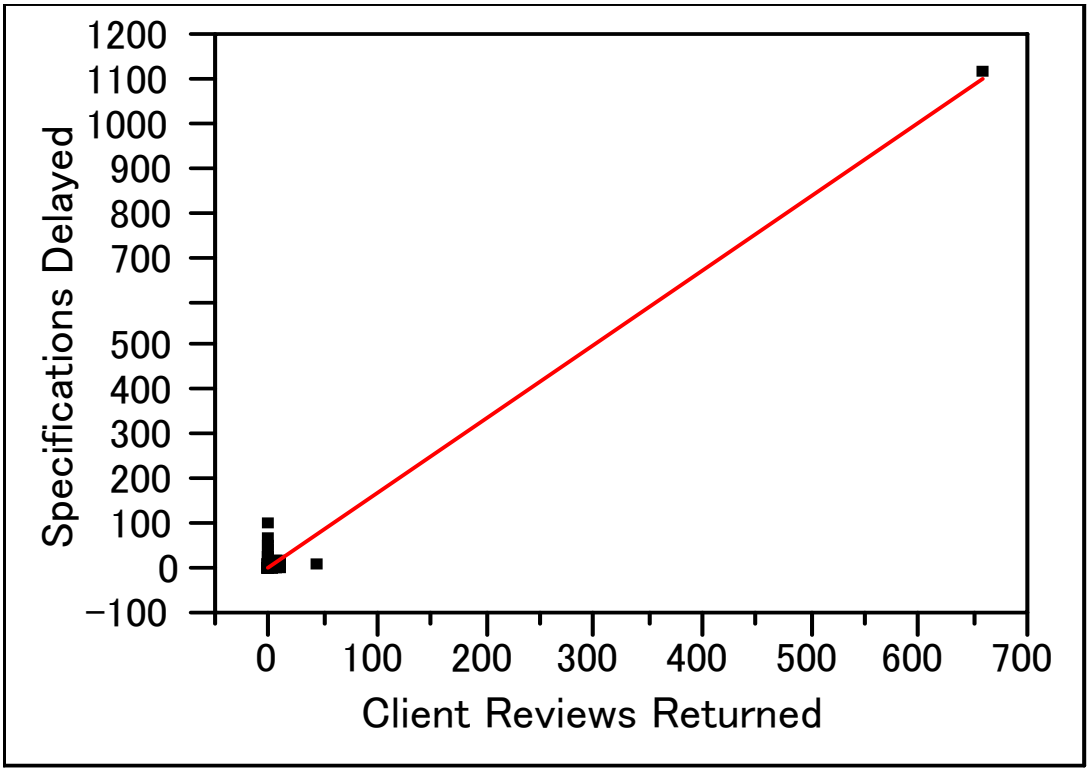

Fig. 4. Scatter plot of the number of client reviews returned vs. the number of delayed specifications

Figure 4 shows a scatterplot of the relationship between the number of client reviews returned (CRR) and the number of specifications delayed (SPD) in the first month. The straight line is the regression line determined using the least-square method. The coefficient of determination, or the square of the correlation coefficient, is 0.98, or nearly equal to 1, indicating that nearly all of the observed points (CRR, SPD) should be on the line. In other words, SPD is almost completely determined by CRR; in fact, the extremely large measurement shown in the upper right-hand corner of the graph is nearly on the line. However, most of the points in the lower left-hand corner (i.e., more than 100 points) are not on or near the line, indicating that even a regression equation with a coefficient of determination of nearly 1 may lead to misleading predictions.

The cause of the confusion is the extreme value in the upper right-hand corner. In medical applications, these are termed outliers and are usually analyzed separately, because those measurements only come from subjects with abnormal conditions. However, in software processes such measurements could represent large profits, and they are not regarded as abnormal. In other words, projects with extremely large measurements should not be removed from a statistical analysis for risk assessment. This is one of the special characteristics of software measurements.

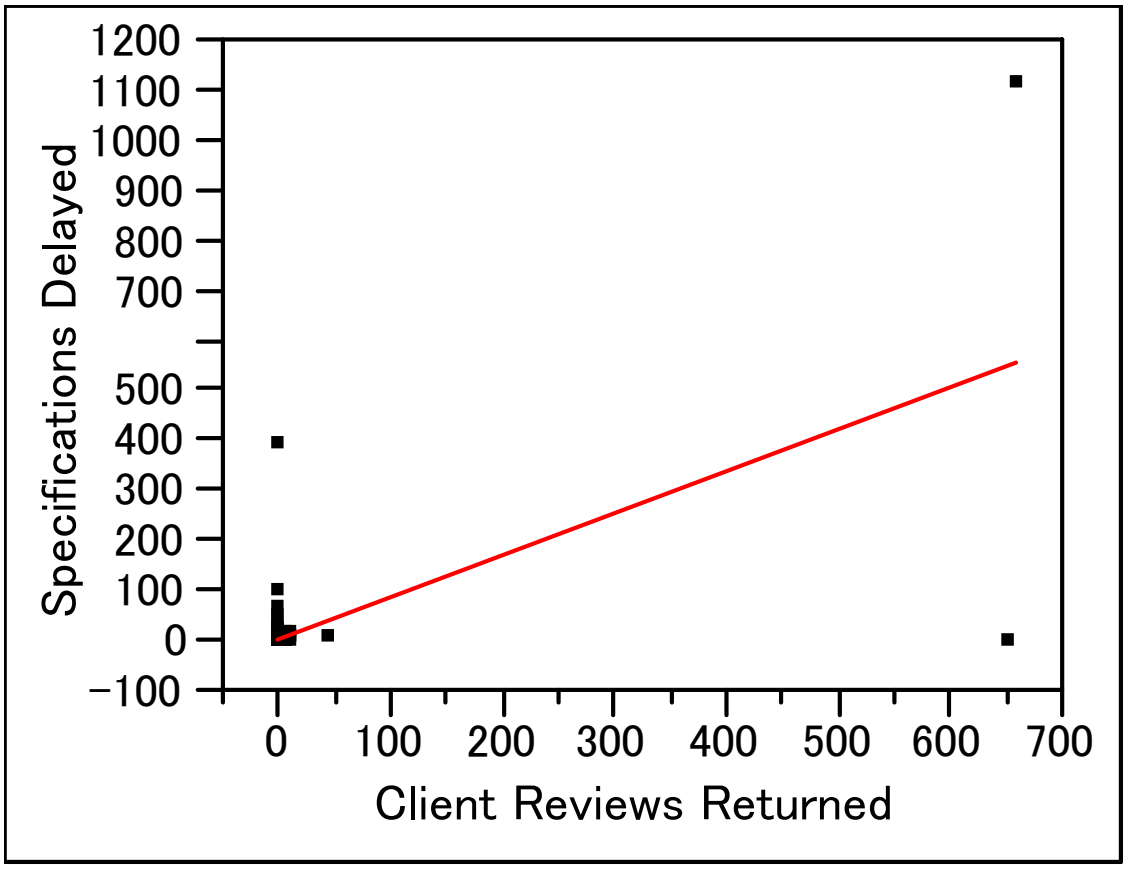

Fig. 5. Scatter plot including three measurements that were excluded in Fig. 4.

Figure 5 shows another example of the problem. The scatterplot, obtained by adding projects excluded in Figure 4 due to missing values in some variables, now shows extreme measurements in the lower right-hand and middle-left regions of the graph. The regression line lies between these extreme points to minimize total deviation. Therefore, even a small number of extreme measurements may diminish the relationship among most measurements and lead to misleading results.

In summary, if all measurements are used in a regression analysis, extreme values may inappropriately influence the relationships among most of the measurements, but if they are excluded, the results will be useless in practice. This dilemma is one of the major problems to be addressed by the software risk assessment and process improvement required by CMM levels 4 and 5. The main objective of this study is to develop a countermeasure to surmount these difficulties and propose a more stable and widely applicable statistical method for risk assessment.

2. Methods

The most important factor of any statistical analysis is the response variable, which essentially determines the practical usefulness of the statistical model and the results. It is typically the most interesting and important variable. For instance, the response variable in cancer clinical trials is usually the survival period after surgery, instead of the reduction ratio of the cancer, because the former is more important than the latter. The response variable chosen by Kitchenham et al. (2004) was:

$$Productivity = Adjusted\ Size/Effort$$

where *Effort* is the estimated size of the project in staff hours and *Adjusted Size* is the total effort to develop an application, measured in staff hours, defined only by those measures

that have a significant relationship with effort so that the expected value is one (see Table 1 of their paper for more details). *Productivity* was chosen as the dependent variable, because the objective of the study was to compare the productivity among countries.

2.1 Profit rate and risk of failure

The aim of our study is to predict the risk of failure of a project at an early stage so that any necessary countermeasures can be taken to reduce the risk. Therefore, we first define the failure of a project based on the profit rate, as follows:

$$Profit\ rate = Amount\ of\ Profit / Amount\ of\ Income$$

There are three types of profit rates: *minimum*, *estimated*, and *actual*. The minimum profit rate is a fixed value determined by the administrator and applied to all projects, normally between 25 and 35%. The estimated profit rate is determined by the project manager for each project when the contract is established, and is usually greater than or equal to the minimum profit rate, but may be lower than that when considering political factors. The actual profit rate is determined at the end of the project according to the above equation. A project is considered to have *failed* if the actual profit rate is less than the minimum profit rate and also below the estimated profit rate. We introduce a variable, *Yore* (literally meaning a *twist*, a technical term used by most Japanese software companies to indicate a

deficit project), defined as *Yore*=1 when a project fails and *Yore*=0 otherwise.

Compared to the productivity used by Kitchenham et al. (2004), which assumes virtually any positive value, the dependent variable in this study is dichotomous, that is, it takes only two values, failure or success. Therefore, the statistical method and the application of the study are completely different from that of Kojima et al (2005) and Kitchenham et al. (2004).

2.2 Data

The following variables are used in the statistical analysis:

UTD1: The number of unit tests delayed in the first month

QAR1: The number of Q&As remaining in the first month

CRR1: The number of client reviews returned in the first month

IRR1: The number of internal reviews returned in the first month

SPD1: The number of specifications delayed in the first month

BOC1: The number of bugs that occurred in the first month

SPC1: The number of specifications changed in the first month

SPA1: The number of specifications added in the first month

GeneTotal: The sum of the scores for eight risk-related items and seven administrative items evaluated by a project manager

LogGT: Logarithmic transformation of GeneTotal

TMDratio: Ratio of CRR1+SPC1 to the Person-Months estimate

Sratio: Ratio of SPD1+ SPC1+SPA1 to the Person-Months estimate

Rratio: Ratio of QAR1+CRR1+IRR1 to the Person-Months estimate

The first eight variables, UTD1–SPA1, were all measurements evaluated by a project manager in the first month after starting the project. GeneTotal is a simple sum of the levels (1, 2, 3, 4, or 5) of eight risk-related items, such as client risk and vendor risk, and eight management-related items, such as the number of staff hours and the estimated size. LogGT is the log-transformation of GeneTotal. TMDratio is a variable proposed by an experienced SE manager as a most effective measurement for predicting the risk of failure. TMD stands for "*Temodori*", a widely used Japanese technical term to indicate "*reworking the same process reluctantly*". Sratio and Rratio represent the rate of troubles in the specifications and reviews, respectively. These variables are reported weekly and summarized every four weeks. Of the 106 projects that participated in RATS, 48 had some missing variables; therefore, 58 projects were used for the following multivariate statistical analysis.

2.3 Statistical methods

The logistic model is applied to determine the most significant variables for distinguishing between successful and failed projects (Moore and McCabe 1998). Since the number of projects available for statistical analysis was small and the number of variables was large, we used both asymptotic tests and exact tests. The asymptotic test applied here was the maximum likelihood method based on the Central Limit Theorem, which is valid with a large sample size, but only an approximation with a small sample size. The exact test was based on non-parametric permutations, and was valid with even a small sample size.

The maximum likelihood method can manage several variables simultaneously, making it particularly suitable when determining the most effective combination of variables among a number of possibilities. Conversely, the exact test can manage only a few variables simultaneously, but the results are precise and reliable. Therefore, the strategy in this study was to screen variables first using the maximum likelihood method, and then confirm the results using the exact test. The maximum likelihood method and exact test were performed using JMP (2006) and LogXact (2006), respectively.

We examined the risk of *Yore* and the relationships between estimated risk and the amount of profit, as follows:

$$Difference = Estimated\ Profit - Actual\ Profit$$

A negative value of *Difference* indicates that the project produced a smaller profit than initially estimated. The projects were divided into six groups according to the estimated risk, and then the average risk and the sum of *Difference* for each group was calculated. Each of the five higher risk groups consisted of ten projects; the lowest risk group had eight projects.

3. Results

3.1 Stepwise logistic analysis

Table 1 shows the significance of each variable based on the scores of the maximum likelihood method. The degree of freedom of the chi-square value was 1 for each variable, and the significant variables included GeneTotal, LogGT, CRR1, SPC1, TMDratio, and Rratio. TMDratio had the highest chi-square value (12.61, p=0.0004), and was the first to be entered into the model. In contrast, Sratio was not significant (p=0.7186).

Table 1. Significance of each variable

Variable	Chi-Square	*P*-value
GeneTotal	**5.97**	**0.0146**
LogGT	4.65	0.031
UTD1	0.03	0.8563
QAR1	3.66	0.0559
CRR1	3.93	0.0473
IRR1	3.78	0.0519
SPD1	3.01	0.0828
BOC1	0.34	0.5609
SPC1	6.87	0.0088
SPA1	0.35	0.5558
TMDratio	**12.61**	**0.0004**
Sratio	0.13	0.7186
Rratio	4.46	0.0346

Table 2. Significance of each variable after entering TMDratio

Variable	Chi-Square	*P*-value
GeneTotal	**3.19**	**0.0739**
LogGT	2.47	0.1163
UTD1	0.26	0.6123
QAR1	0.68	0.4104
CRR1	0.15	0.6981
IRR1	0.13	0.7204
SPD1	0	0.9808
BOC1	0.13	0.7159
SPC1	0.24	0.6248
SPA1	0.15	0.702
TMDratio	**5.29**	**0.0214**
Sratio	0.03	0.8592
Rratio	0.74	0.3883

Table 2 shows the significance of each variable as a pair with TMDratio. The chi-square values denote the contribution of each variable independently of TMDratio. None of them was significant. Only GeneTotal approached significance (p=0.074). Since the sample

size was small and the results approximate, we incorporated it into the model.

Table 3. Final results of the stepwise variable selection

Variable	Estimate	Chi-square	*P*-value
GeneTotal	**0.0304**	**2.95**	**0.0857**
LogGT		0	0.9649
UTD1		0.28	0.5976
QAR1		0	0.9439
CRR1		0.55	0.4582
IRR1		0.42	0.5151
SPD1		0.23	0.6279
BOC1		0.18	0.6677
SPC1		0.18	0.6695
SPA1		0.03	0.8634
TMDratio	**1.72**	**4.76**	**0.0291**
Sratio		0.06	0.8008
Rratio		0.06	0.8134

Table 4. Exact *p*-values and estimates

	Variable	Estimate	*P*-value
(a)	TMDratio	1.7287	0.0012
(b)	GeneTotal	0.0309	0.0155
(c)	TMDratio	1.3841	0.0311
	GeneTotal	0.0305	0.0797
(d)	Sratio	0.0111	0.4485
(e)	Rratio	0.3374	0.0718

Table 3 shows the final results of the stepwise logistic analysis. Table 4 shows the exact significance of (a) TMD ratio, (b) GeneTotal, (c) the two variables together. For reference, the exact P-values of Sratio and Rratio were also tabulated in (d) and (e), respectively. The p-values of the two variables, TMD ratio (0.0291) and GeneTotal (0.0857), and the estimate of the regression coefficient for GeneTotal (0.0304) were

similar to those shown in Table 3. The estimate of the regression coefficient for TMDratio (1.3841) was, however, considerably different from that in Table 3 (1.72).

3.2 Goodness of fit

Table 5. Goodness of fit of the logistic model

Estimated probability	Failure (Yore = 1)			Success (Yore = 0)			
	Obs.	Expected	Cumulat.	Obs.	Expected	Cumulat.	Total
<0.039	0	0.1538	0.0000	5	4.8462	0.1538	5
<0.052	0	0.2279	0.0000	5	4.7721	0.3817	5
<0.104	1	0.5276	1.0000	5	5.4724	0.9092	6
<0.113	0	0.6536	1.0000	6	5.3464	1.5628	6
<0.149	2	0.7734	3.0000	4	5.2266	2.3362	6
<0.174	1	0.9529	4.0000	5	5.0471	3.2891	6
<0.220	0	1.2325	4.0000	6	4.7676	4.5216	6
<0.300	3	1.6275	7.0000	3	4.3725	6.1491	6
<0.418	2	2.2333	9.0000	4	3.7667	8.3824	6
<0.999	4	4.6176	13.0000	2	1.3824	13.0000	6
Total	13	13		45	45		58

"Obs." and "Cumulat." stand for observed and cumulative frequencies, respectively.

Table 5 shows a goodness-of-fit of the logistic model using the two variables TMDratio and GeneTotal as covariables. First, the probability of failure, or the probability of *Yore* = 1, was calculated for each project using the logistic equation with the estimated regression coefficients for the two variables (Moore and McCabe, 1998; Hosmer and Lemeshow, 1989). The probability was referred to as the *estimated probability*. Then, the 58 projects were sorted by the estimated probability, from the lowest to the highest. In the

table, Observed Yore=1 includes the 13 failed projects, and Observed Yore=0 included the 45 successful ones. The sum of the estimated probabilities of the smallest five projects was 0.1538, which was regarded the expected number of the *Yore*=1 projects. The difference, 5 – 0.1538 = 4.8462, is the expected number of *Yore*=0 projects. These figures are shown in the first row. The sum of the estimated probabilities of the highest six projects was 4.6176 (shown in the penultimate row), with a difference of 6 – 4.6176 = 1.3824. The other rows were calculated similarly.

Table 6. Average risk and the sum of *Difference* between the estimated and the actual profit.

Ave. Risk	Difference
0.62	-39934
0.27	-38845
0.17	-8245
0.12	8037
0.08	24407
0.04	5596

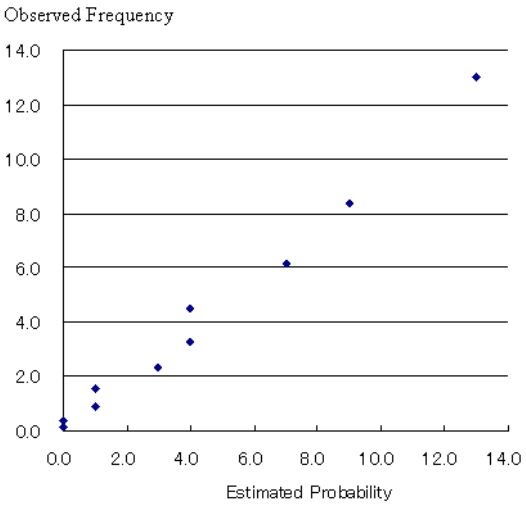

Fig. 6. Cumulative estimated probability vs. the cumulative observed frequency of Yore=1

Figure 6 shows the scatterplot of the cumulative estimated probability vs. the cumulative observed frequency of Yore=1. The estimated and observed risk values agreed well with each other, indicating the appropriateness of the model.

3.3 Profit by risk level

Table 6 tabulates the average risk and the sum of *Difference* for the projects by risk. The highest risk group had an average risk of 0.62 and a sum of *Difference* of –39934. Figure 7 is a bar chart representation of Table 6. It shows clear relationships between the estimated risk and the profit or loss.

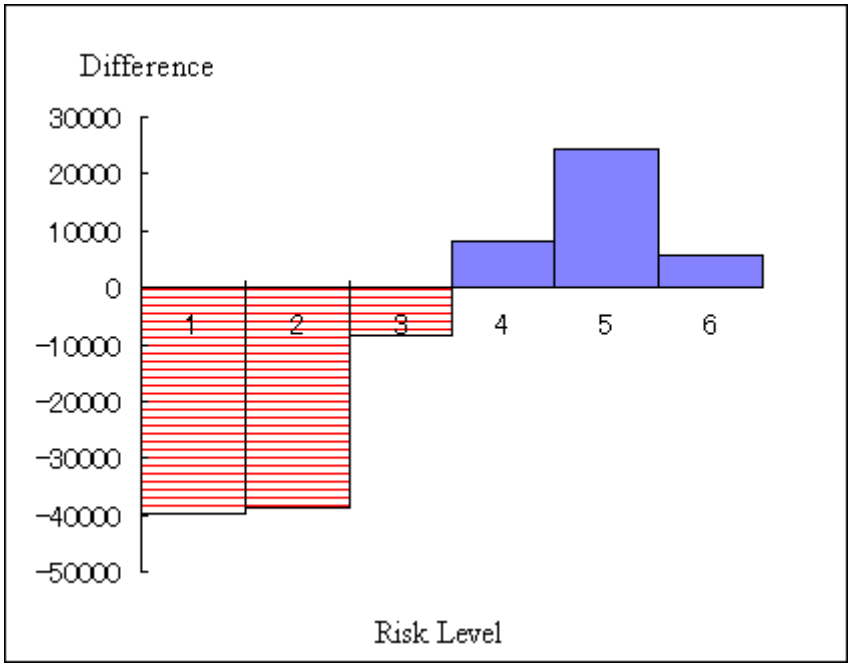

Fig. 7. Bar chart presentation of Table 6. The highest risk group is denoted by "1," the next highest by "2," and the lowest by "6."

4.Discussion

4.1 Statistical analysis

One of our objectives was to develop a sound statistical method for evaluating risks that can accommodate both normal and large-sized projects simultaneously, since the linear regression method is not very reliable in some situations. Outliers in a regression analysis (in this case, large-sized projects), are influential factors such that the removal of them from the data set would significantly change the relationships among the endpoint and the explanatory variables.

Some tools to identify outliers in a linear regression analysis, such as Cook's distance, are available in most statistical software (JMP 2006). To accommodate outliers in a linear regression analysis requires special care (Barnett 1998), which may not always be applicable. To simply reduce the influence of the individual measurements, we proposed transforming the continuous endpoint, such as profit, to a binary variable such as "Yore." Then statistical analyses present an estimated risk of failure, instead of profit, for each project.

4.2 Prognostic factors

TMDratio was the most significant prognostic variable for predicting the risk of failure. GeneTotal was also significant as a single variable, but only marginally significant after TMDratio is entered in the model. GeneTotal is a metric for quantifying the risk of failure in a subjective manner determined by the manager at the start of the project. By contrast, TMDratio is determined one month after project initiation. The fact that TMDratio was still highly significant after GeneTotal was first entered indicates that TMDratio has a significant contribution to failure risk, independent of GeneTotal. In other words, the value of TMDratio may not be completely predictable at the start of a project because it is partly caused by events that occur after the start of the project.

TMDratio is the ratio of the sum of CRR1 (the number of client reviews returned in the first month) and SPC1 (the number of specifications changed in the first month) to the Person-Months estimate. Since both CRR1 and SPC1 are strongly linked to the clients, the results provide some evidence that the loss of profit of failed projects with a high TMDratio may be partly attributable to clients.

Since GeneTotal is marginally significant even jointly with TMDratio, it would lower the risk of failure to take a preventive measure to reduce the GeneTotal value before the start

of a project. GeneTotal is defined as the sum of 15 variables. Therefore, to determine the most effective measure, all 15 variables should be analyzed independently to identify the most significant variables among them, using the same statistical analysis. However, this analysis would require a much larger sample size for reliable results.

Sratio was not significant, which is surprising because the design specification is typically the most important to process control. In fact, the chi-square value of Sratio was less than that of any of its components, SPD1, SPC1, and SPA1, indicating that the three variables should be treated independently in risk analyses for failure prediction. Rratio was also only marginally significant in the exact test. These unexpected negative results suggest that finding effective prognostic variables requires more accumulation of data from various application fields to provide a more comprehensive and deeper understanding of the process.

4.3 Feedback to managers

It was discussed that effective countermeasures would have been possible to save some of the higher risk projects at their initial stages, that would have reduced the overall loss. The results were sent back to the managers who submitted raw data. To apply the results

obtained from the statistical analysis to a new project, we would first estimate the risk of *Yore* for each project using the logistic equation to identify those at higher risk. Then, any effective measure could be performed to reduce risk in an early stage. The risk should be re-evaluated using data collected after the treatment. If the risk is still too high, still other measures should be considered. The strategy is partly implemented in RATS and under further investigation.

5. Conclusion

To identify and track risks in order to detect and resolve problems in the earlier stages of the software development process, Software Research Associates (SRA) began the so-called “the risk assessment and tracking system” (RATS) in 1994. To implement the idea and to improve its ability to perform relevant prognostication, we developed a list of critical prognostic variables to distinguish successful and failed projects based on a comprehensive statistical analysis of the accumulated data in RATS. The endpoint *Yore* is introduced to indicate a deficit project in terms of the profit. Since the linear regression method is not very reliable with extremely large projects, to evaluate the risk of failure we developed a sound statistical method that can accommodate both normal-sized and large projects simultaneously. To reduce the influence of the individual measurements, we

proposed transforming a continuous endpoint, such as profit, into a binary endpoint, such as *Yore*. The logistic model found the TMD ratio to be the most effective variable for distinguishing between the two endpoints. TMD stands for "*temodori*", a widely used Japanese technical term that means "*reworking the same process reluctantly*". The next useful variable was GeneTotal, defined as the sum of the levels of eight risk-related items and eight management-related items. The statistical analyses present an estimated risk of failure, instead of profit, for each project.

Our results should be interpreted with caution because this is an exploratory study with a small sample size. If a large number of projects were available for analysis, it would be interesting to apply our method independently to groups of different size (*i.e.*, small, intermediate, and large), and see if it produces similar results. If not, some amendments to adjust for project size should be developed to improve the applicability of our method.

References

Amasaki S., Yoshitomi T., Mizuno O., Takagi Y., and Kikuno T. (2005) A new challenge for applying time series metrics data to software quality estimation, *Software Quality Journal*, **13**: 177–193.

Barnett, V. (1998) Multivariate outliers, in *Encyclopedia of Biostatistics* **4** (ed. Armitage, P. and Colton, T.), Wiley, New York 2920-2926.

Bieman J. 2004. The role of prognostication in software design, *Software Quality Journal*, **12**: 7–8.

De Lacalle, L. N. L., Lamikiz, A. Salgado, M. A., Herranz, S., Rivero, A. (2002) Process planning for reliable high-speed machining of moulds. *International Journal of Production Research* **40** 2789-2809.

Ellis, R.C.T., Wood, G.D. and Thorpe, T. (2004). Technology-based learning and the project manager. Engineering, Construction and Architectural Management 11, 358-365

Hirayama, M., Mizuno, O. and Kikuno, T. (2004) Test item prioritizing metrics for selective software testing. IEICE Transactions on Information and Systems E87-D, 2733-2743

Hosmer, D.W. and Lemeshow, S. (1989) *Applied Logistic Regression*, New York, Wiley.

Huang, C.-Y. (2005). Performance analysis of software reliability growth models with testing-effort and change-point. Journal of Systems and Software **76**, 181-194

JMP statistics software. (2006) SAS Institute, Cary, NC, USA. http://www.jmp.com/

Ingham, M.D., Rasmussen, R.D., Bennett, M.B., Moncada, A.C. (2005) Engineering

complex embedded systems with state analysis and the mission data system, *Journal of Aerospace Computing, Information and Communication* **2**, 507-536.

Kang, B., Kwon, Y.-J. and Lee, R.Y. (2005) A design and test technique for embedded software. *Proceedings - Third ACIS International Conference on Software Engineering Research, Management and Applications* **SERA 2005** No. 1563157, 160-165.

Kitchenham, B. and Mendes, E. (2004) Software Productivity Measurement Using Multiple Size Measures, *IEEE Trans Software Engineering*, **30**, 1023-1035

Khoshgoftaar, T.M, Seliya, N. and Sundaresh, N. (2006) An empirical study of predicting software faults with case-based reasoning, *Software Quality Journal* **14**, 85-111.

Khoshgoftaar, T.M, Szabo, R.M. and Woodcock, T.G.（1994） An empirical Study of program quality during testing and maintenance, *Software Quality Journa*l **3**, 137-151.

Kitchenham, B. and Mendes, E. (2004) Software Productivity Measurement Using Multiple Size Measures, *IEEE Trans Software Engineering*, **30**, 1023-1035.

Kojima T., Hasegawa T., and Nakamura T. (2005) *Proceedings of SEPG Japan* (in Japanese) available in CD format.

Liu, F., Noguchi, K., Dhungana, A., Srirangam, V. V. N. S. N. A. and Inuganti, P. (2006) A quantitative approach for setting technical targets based on impact analysis in software quality function deployment (SQFD) , *Software Quality Journal* **14**, 113-134.

LogXact statistical software. (2006) Cytel Statistics and Epidemiology Research Corporation, Seattle, WA, USA. http://www.scientific-solutions.ch/tech/logxact/index.html

Maydl, W. (2004) Model checking for component-based software development for embedded systems. *Proceedings of the Eighth IASTED International Conference on Software Engineering and Applications* 331-338

Moore D.S. and McCabe G.P. (1998) *Introduction to the Practice of Statistics*, New York, W.H. Freeman & Company.

Pfahl, D., Laitenberger, O., Dorsch, J. and Ruhe, G. (2003) An Externally Replicated Experiment for Evaluating the Learning Effectiveness of Using Simulations in Software Project Management Education, *Empirical Software Engineering* **8**, 367-395

Williams M. (2005) Software glitch halts Tokyo Stock Exchange, *COMPUTERWORLD* November 01. http://www.computerworld.com/action/article.do?command=viewArticleBasic&articleId=105838