Prediction of Defect Distribution based on Project Characteristics for Proactive Project Management

Youngki Hong, Wondae Kim, Jeongsoo Joo
SK C&C Ltd.
SK u-Tower, 25-1 Jeongja-dong, Bundang-gu, Seongnam-si, Gyeonggi-do
Republic of Korea
{ young, windy6471, jsjoo } @ SKCC.COM

ABSTRACT

As software has been pervasive and various software projects have been executed since the 1970's, software project management has played a significant role in software industry. There are three major factors in project management; schedule, effort and quality. Especially, to represent quality of products, there are various possible quality characteristics of software, but in practice, frequently, quality management revolves around defects, and delivered defect density has become the current de facto industry standard. The researches related to software quality have been focused on modeling residual defects in software in order to estimate software reliability.

However, only the predicted number of defects cannot be sufficient information to provide basis for planning quality assurance activities and assessing them during execution. That is, in order to let projects managers be able to identify the project related information in early phase, we need to predict other possible information for assuring software quality such as defect density by phases, defect types and so on. In this paper, we propose a new approach for predicting distribution of in-process defects, their types based on project characteristics in early phase. For this approach, the model for prediction is established using the curve fitting method and the regression analysis. The maximum likelihood estimation is used in fitting the Weibull probability density function to the actual defect data, and the regression analysis is used to identify the relationship between the project characteristics and the Weibull parameters. The research model is validated by using cross-validation technique.

Categories and Subject Descriptors

D.2 [**Software Engineering**]: Management; K.6.3 [**Management of Computing and Information Systems**]: Software Management— *Software process*

General Terms

Management, Measurement, Reliability.

Keywords

Project Management, Software Reliability, In-process Defect Prediction, Defect Distribution, Weibull Distribution Function, Maximum Likelihood Estimation

1. INTRODUCTION

Customer satisfaction has become the major objective of many organizations attempting to survive and thrive in today's increasingly competitive world. Most software organizations are trying to research to have the ability to develop and deliver reliable, usable software within budget and schedule commitments. Usually, a mature organization possesses an organization-wide ability for managing software development and maintenance processes. Project managers in mature organizations monitor the quality of the software products and the process that produces them based on an objective, quantitative basis for judging product quality [1].

For better project management, one of the most important objectives of the software engineering community has been to develop useful models that can explain the software development life-cycle and accurately predict the cost, schedule and quality of developing a software product [2][3]. Especially, to represent quality of products, there are various possible quality characteristics of software, and there is even an international standard for this [2]. In practice, frequently, quality management revolves around defects, and delivered defect density, a number of defects per unit size in the delivered software, has become the current de facto industry standard [4][5]. Therefore, the prediction of software defects, i.e., deviations from specifications or expectations, has been an important research topic in the field of software engineering. So far, many efforts have been concentrated specifically in predicting the number of defects in the system, estimating the reliability of the systems as statistical functions to time-to-failure, and understanding the importance of design and testing processes on defect counts [6]. However, only the number of defects cannot be sufficient information to provide basis for planning quality assurance activities and assessing them during execution. That is, for project management to be improved, we need to predict other possible information of software quality such as in-process defects, their types and so on.

Our research is motivated by the aim of providing additional quality information to project manager and members in the early phase for improved project plan and execution. This will help us plan suitable quality assurance activities and prevent possible defects in projects. It can assist us to reduce the efforts of performing reworks in software development projects. Thus, it will also help in the reduction of redundant quality assurance activities and the cost of producing high quality software. In this research, we suggest a predictive defect type distribution model based on the software project characteristics in the early stage.

The rest of paper is organized as follows. In section 2, our modeling approach is described. In section 3, the data is presented to use in constructing the proposed model. In section 4, the process of constructing the model and the defect type prediction model are presented. Section 5 presents the validation of our proposed approach. In section 6, some related works are described. Lastly, the conclusion and future works are drawn.

2. RESEARCH APPROACH

This section describes the predictive defect type distribution model and the approach to construct it.

2.1. Overview of the Proposed Model

Our research is aimed to predict the distribution of in-process defects, their types to be detected in the software project. The Figure 1 shows the overall view of the predictive defect distribution model.

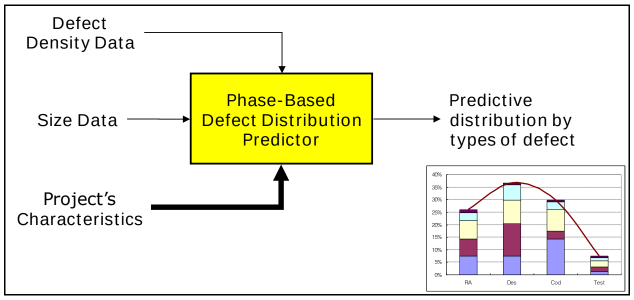

Defect Density Data
Size Data
Project's Characteristics
Phase-Based Defect Distribution Predictor
Predictive distribution by types of defect

Figure 1. Overview of the proposed model

2.2. Approach

The modeling approach outlines construction of the proposed defect distribution prediction model shown in Figure 2. There are 7 steps in modeling approach: 1) existing literature analysis, 2) behavioral analysis, 3) defect data gathering, 4) statistical modeling, 5) regression analysis, 6) model validation and 7) data accumulation for refining the model in the future. The steps and their explanations are as following.

1) Existing literature analysis: The first step in developing a software estimation/prediction model is to determine the factors (or predictor variables) that affect the software attribute being estimated (i.e. the response variable). This can be done by reviewing existing literature and analyzing the influence of parameters on the response variable [14].

2) Behavioral analysis: Once the parameters have been determined, a behavioral analysis should be carried out to understand the effects of each of the parameters on the response variable. That is, this shows the behavioral effects of higher vs. lower levels of each factor on project quality levels [14].

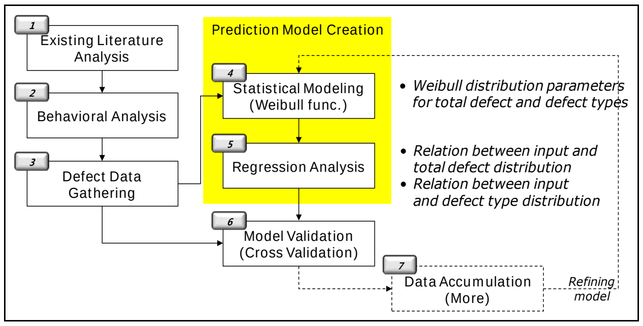

1 Existing Literature Analysis
2 Behavioral Analysis
3 Defect Data Gathering
Prediction Model Creation
4 Statistical Modeling (Weibull func.)
5 Regression Analysis
6 Model Validation (Cross Validation)
7 Data Accumulation (More)
• Weibull distribution parameters for total defect and defect types
• Relation between input and total defect distribution
• Relation between input and defect type distribution
Refining model

Figure 2. Modeling approach

3) Defect data gathering: After a thorough study of the behavioral analyses was done, the characteristics of projects and defect data have to be gathered according to the results of behavioral analysis. At this time, the defect data should include defects at all phases of the development cycle. Also, the more detail phases for differentiating detected defects, the more accurate output we will obtain. We use the existed project characteristics and the defect data in the organization.

4) Statistical modeling: Using the actual project data, curve fitting is performed to extract the parameters of the Weibull distribution function per each project. These values will be input data for regression analysis.

5) Regression analysis: In regression analysis, we identify relationship between historical project characteristics and parameters of defect distribution from the statistical modeling.

6) Model validation: After statistical modeling and regression analysis, we obtain various coefficients to be able to predict output. To validate the results of the model, we perform the cross-validation because we have little project data to do the test-set validation. Cross-validation is an approach to estimate how well the model we've made from some training data is going to perform on future data.

7) Data accumulation (more) for refining the model: We can continue to gather data, and refine the model with them to be more reliable one.

3. Data Description

To construct a defect type distribution model, the actual software development project data are needed to analysis. The required data are the characteristics of the projects and their defect data detected in the software development life cycle.

3.1. Project Characteristics

For this model construction, we gathered data from 18 completed software development projects which were performed in public business sector (government, military, etc.) from 2003 to 2006. There can be lots of quantitative and qualitative characteristics of differentiating software projects in each organization [3][18]. In our case, the 10 factors of project characteristics are included in three categories: project basic information, human resource information and development environment information. They are the existed information for differentiating projects in the organization. The factors and their brief explanations in each category are as following Table 1.

Table 1. Project characteristics for the model

Category/Charac.	Explanation
Project basic information	
Project size	Value of function point to be developed in the project
Project duration	Total months of the project duration
Development solution	Type of solution (product) to be developed in the project (MIS, Web page, etc.)
Industry area	Industry area that the system is applied in
Development methodology	Type of development methodology used in the project (IE, Web, etc.)
Human resource information	
Number of project members	Number of project members (dissociated from our own members and subcontracted members)
Productivity	Project productivity which is function points divided by number of person month
Experience and capability	Overall experience and capability about project and similar application development (Special, High, Med, Low)
Development environment information	
Development language	Type of development language used in the project (C/C^{++}, JAVA, PwrBld, etc.)
DBMS type	Type of DBMS used in the project (Oracle, Sybase, MySQL, etc.)

3.2. Defect Data

In addition to project's characteristics, we also use defect data obtained from all phases in development life cycle (requirement analysis, design, coding and testing) for the software projects. The defect data were acquired from requirement review, design review, code inspection and test by project members and testers.

The defect data have been recorded according to the organizational standard defect types when project members detected them in each phase of the projects. There are five major types of defects in the gathered data: consistency, function, standard, performance and miscellaneous. Table 2 shows the summary of defect types.

Table 2. Summary of defect types

Type of defect	Explanation
Consistency (C)	Defects to have little consistency between the previous artifacts and the current artifact
Function (F)	Defects to affect the functionality due to incorrect functional explanation or wrong algorithm or data structure, etc.
Standard (S)	Defects to have little observance to the rules such as customer's standard, project standard methodology, coding rules, etc.
Performance (P)	Defects to impact the performance due to incorrect design or inefficient algorithm or data structure, etc.
Miscellaneous (E)	Defects not to be categorized to above types

The defect type distributions per development phase can be modeled according to the Weibull distribution function because they follow the characteristics of the Weibull distribution (decreasing monotonously after having one peak) shown in Figure 3.

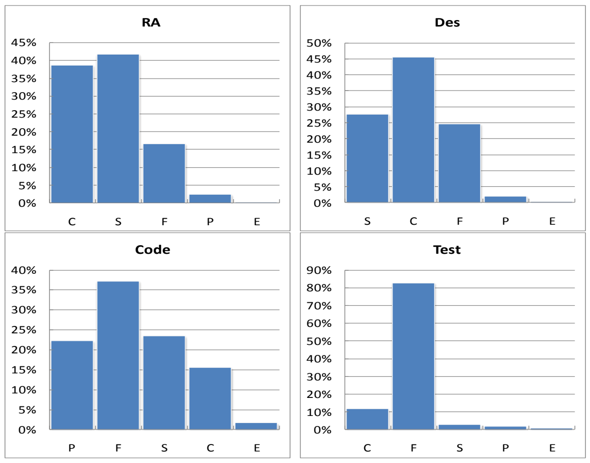


Figure 3. Average defect type distribution in each phase

4. Model Construction

We construct the proposed model with historical project information and defect data using statistical modeling and regression analysis. In the statistical modeling, we use the maximum likelihood estimation technique for fitting the Weibull distribution curve to the historical defect data. In the regression analysis, we determine the coefficients about the relationship between project characteristics and the Weibull parameters of defect distribution.

4.1. Weibull Distribution Function

The Weibull distribution, firstly introduced in 1939 by W. Weibull, is defined by the following probability density function and cumulative distribution function.

$$\text{Weibull PDF : } f(x,\alpha,\beta)=\left(\frac{\alpha}{\beta^{\alpha}}\right)\cdot x^{(\alpha-1)}\cdot e^{-\left(\frac{x}{\beta}\right)^{\alpha}}$$

$$\text{Weibull CDF: } F(x,\alpha,\beta)=1-e^{-\left(\frac{x}{\beta}\right)^{\alpha}}$$

The purpose of the Weibull method is to characterize the failure distribution and to make inferences about the failure mode. It can be used with relatively small sample sizes [15]. A simple, flexible method for modeling continuous event times is to assume they follow a Weibull distribution with shape parameter, α, and scale parameter, ß, which are positive numbers.

The Weibull function can be fitted to the actual data by maximum likelihood (ML) that is widely used in engineering applications. The parameter estimates can be used to identify other characteristics of the distribution [16].

4.2. Maximum Likelihood Estimation

Maximum likelihood estimation (MLE) is a popular statistical method used to calculate the best way of fitting a mathematical model to some data. Modeling real world data by estimating maximum likelihood offers a way of tuning the free parameters of the model to provide an optimum fit [17].

Commonly, one assumes that the data drawn from a particular distribution are independent, identically distributed with unknown parameters. This considerably simplifies the problem because the likelihood can then be written as a product of n univariate probability densities:

$$L(\theta) = f_\theta(x_1 | \theta) f_\theta(x_{2i} | \theta) \ldots f_\theta(x_n | \theta) = \prod_{i=1}^{n} f_\theta(x_i | \theta)$$

and since maxima are unaffected by monotone transformations, one can take the logarithm of this expression to turn it into a sum:

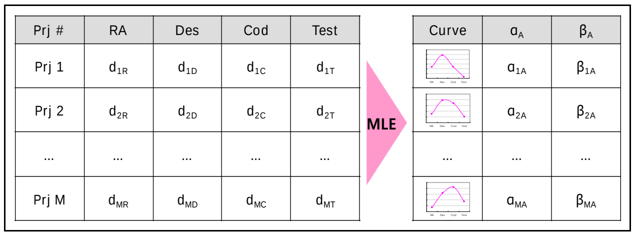
$$\ln L(\theta) = \sum_{i=1}^{n} \log f_\theta(x_i | \theta)$$

In consequence, according to above derived equations, we can select parameters to make the value 0 of ln L(θ).

4.3. Statistical Modeling

To perform the statistical modeling is to estimate the Weibull distribution parameters from actual data. Besides, we have to perform statistical modeling from the entire defect distribution per project to each defect type distribution per development phase.

First, for performing estimation of the entire defect distribution per project, we need to arrange the number of defects by all phases of software development life cycle. The defects in entire phases can be modeled according to the Weibull distribution because the distribution of defects follows the characteristics of the Weibull distribution. Then, the estimation of the parameters per each project can be performed using the maximum likelihood estimation (MLE). The Figure 4 shows the overview of estimation of the Weibull parameters for the entire defect distribution per each project.



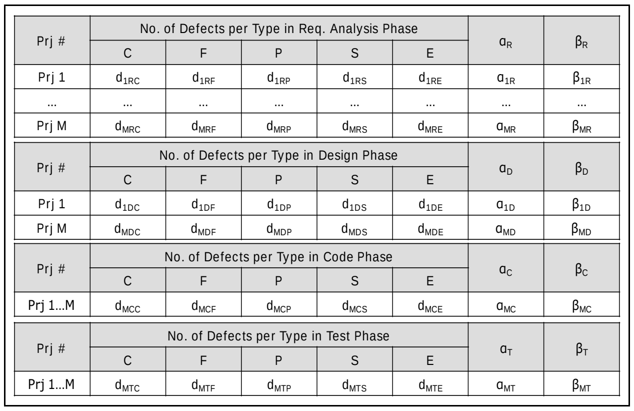
Prj #	RA	Des	Cod	Test
Prj 1	d_{1R}	d_{1D}	d_{1C}	d_{1T}
Prj 2	d_{2R}	d_{2D}	d_{2C}	d_{2T}
…	…	…	…	…
Prj M	d_{MR}	d_{MD}	d_{MC}	d_{MT}

Curve	α_A	β_A
	α_{1A}	β_{1A}
	α_{2A}	β_{2A}
…	…	…
	α_{MA}	β_{MA}

Figure 4. Parameter estimation of entire defect dist.

After that, we perform an estimation of each defect type distribution per development phase. For doing so, we arrange defect data by defect types in each phase per each project. Also, the defect type distributions per development phase can be modeled with the Weibull distribution function because they follow the characteristics of the Weibull distribution.

Through the statistical modeling, we can obtain 10 Weibull parameters for each project: α_A, β_A, α_R, β_R, α_D, β_D, α_C, β_C, α_T, β_T. This set of parameters is used to determine the relationship with the project characteristics. The Figure 5 shows the overview of estimation of the Weibull parameters for the defect type distribution per each phase.



Prj #	No. of Defects per Type in Req. Analysis Phase					α_R	β_R
	C	F	P	S	E		
Prj 1	d_{1RC}	d_{1RF}	d_{1RP}	d_{1RS}	d_{1RE}	α_{1R}	β_{1R}
…	…	…	…	…	…	…	…
Prj M	d_{MRC}	d_{MRF}	d_{MRP}	d_{MRS}	d_{MRE}	α_{MR}	β_{MR}

Prj #	No. of Defects per Type in Design Phase					α_D	β_D
	C	F	P	S	E		
Prj 1	d_{1DC}	d_{1DF}	d_{1DP}	d_{1DS}	d_{1DE}	α_{1D}	β_{1D}
Prj M	d_{MDC}	d_{MDF}	d_{MDP}	d_{MDS}	d_{MDE}	α_{MD}	β_{MD}

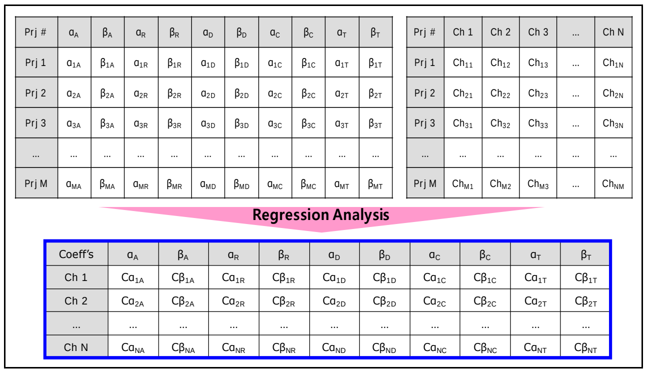
Prj #	No. of Defects per Type in Code Phase					α_C	β_C
	C	F	P	S	E		
Prj 1…M	d_{MCC}	d_{MCF}	d_{MCP}	d_{MCS}	d_{MCE}	α_{MC}	β_{MC}

Prj #	No. of Defects per Type in Test Phase					α_T	β_T
	C	F	P	S	E		
Prj 1…M	d_{MTC}	d_{MTF}	d_{MTP}	d_{MTS}	d_{MTE}	α_{MT}	β_{MT}

Figure 5. Parameter estimation of defect type dist.

4.4. Regression Analysis

Through performing regression analysis, we can identify the relationship between the project characteristics and the 10 parameters about 5 Weibull distributions. Thus, we can obtain the coefficients to use predicting defect distribution for new projects. The Figure 6 shows the overview of regression analysis between project characteristics and the parameters of Weibull distributions.



Prj #	α_A	β_A	α_R	β_R	α_D	β_D	α_C	β_C	α_T	β_T
Prj 1	α_{1A}	β_{1A}	α_{1R}	β_{1R}	α_{1D}	β_{1D}	α_{1C}	β_{1C}	α_{1T}	β_{1T}
Prj 2	α_{2A}	β_{2A}	α_{2R}	β_{2R}	α_{2D}	β_{2D}	α_{2C}	β_{2C}	α_{2T}	β_{2T}
Prj 3	α_{3A}	β_{3A}	α_{3R}	β_{3R}	α_{3D}	β_{3D}	α_{3C}	β_{3C}	α_{3T}	β_{3T}
…	…	…	…	…	…	…	…	…	…	…
Prj M	α_{MA}	β_{MA}	α_{MR}	β_{MR}	α_{MD}	β_{MD}	α_{MC}	β_{MC}	α_{MT}	β_{MT}

Prj #	Ch 1	Ch 2	Ch 3	…	Ch N
Prj 1	Ch_{11}	Ch_{12}	Ch_{13}	…	Ch_{1N}
Prj 2	Ch_{21}	Ch_{22}	Ch_{23}	…	Ch_{2N}
Prj 3	Ch_{31}	Ch_{32}	Ch_{33}	…	Ch_{3N}
…	…	…	…	…	…
Prj M	Ch_{M1}	Ch_{M2}	Ch_{M3}	…	Ch_{NM}

Coeff's	α_A	β_A	α_R	β_R	α_D	β_D	α_C	β_C	α_T	β_T
Ch 1	$C\alpha_{1A}$	$C\beta_{1A}$	$C\alpha_{1R}$	$C\beta_{1R}$	$C\alpha_{1D}$	$C\beta_{1D}$	$C\alpha_{1C}$	$C\beta_{1C}$	$C\alpha_{1T}$	$C\beta_{1T}$
Ch 2	$C\alpha_{2A}$	$C\beta_{2A}$	$C\alpha_{2R}$	$C\beta_{2R}$	$C\alpha_{2D}$	$C\beta_{2D}$	$C\alpha_{2C}$	$C\beta_{2C}$	$C\alpha_{2T}$	$C\beta_{2T}$
…	…	…	…	…	…	…	…	…	…	…
Ch N	$C\alpha_{NA}$	$C\beta_{NA}$	$C\alpha_{NR}$	$C\beta_{NR}$	$C\alpha_{ND}$	$C\beta_{ND}$	$C\alpha_{NC}$	$C\beta_{NC}$	$C\alpha_{NT}$	$C\beta_{NT}$

Figure 6. Regression analysis for prediction parameters

4.5. Proposed Model

After modeling, now we can construct a predictive defect type distribution model. The constructed model is formed as a discrete Weibull distribution function as bellow equation, where K is total defect density, i is phase sequence, j is type of defect, α and β are shape parameter and scale parameter for defect type distribution of each phase.

$$P(i, j) = K_i \cdot \left(\frac{\alpha_i}{\beta_i^{\alpha_i}} \cdot j^{(\alpha_i - 1)} \cdot e^{-\left(\frac{j}{\beta_i}\right)^{\alpha_i}} \right)$$

$$= K \cdot \left(\frac{\alpha_A}{\beta_A^{\alpha_A}} \cdot i^{(\alpha_A - 1)} \cdot e^{-\left(\frac{i}{\beta_A}\right)^{\alpha_A}} \right) \cdot \left(\frac{\alpha_i}{\beta_i^{\alpha_i}} \cdot j^{(\alpha_i - 1)} \cdot e^{-\left(\frac{j}{\beta_i}\right)^{\alpha_i}} \right)$$

Our research is aimed to predict the distribution of in-process defects, their types to be detected in the software project. The Figure 7 shows the overall view of the predictive defect distribution model in more detail.

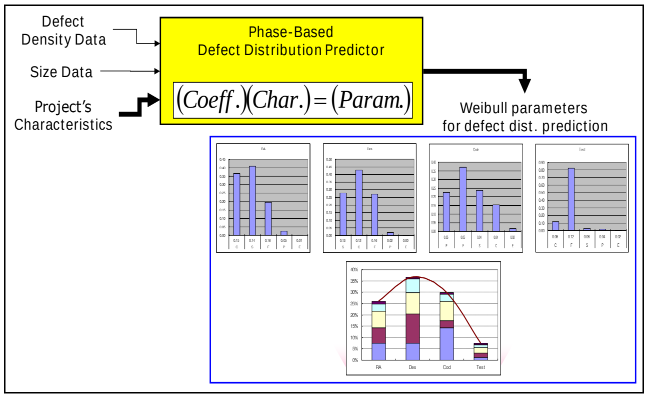

Defect Density Data
Size Data
Project's Characteristics
Phase-Based Defect Distribution Predictor
(Coeff.)(Char.)=(Param.)
Weibull parameters for defect dist. prediction
RA
Des
Cod
Test

Figure 7. Overview of the proposed model in more detail

In order to acquire the Weibull parameters for a new project, we perform matrix calculation between the coefficients matrix that we obtained in modeling and the characteristics vector of a new project that will be initiated, where Cα's and Cβ's are the coefficients for prediction, Ch's are characteristics of a new project, α's and β's are the estimated Weibull parameters, and ε is an error term on each characteristic.

$$\begin{pmatrix} C\alpha_{1A} & C\alpha_{2A} & \cdots & C\alpha_{N-1A} & C\alpha_{NA} \\ C\beta_{1A} & C\beta_{2A} & \cdots & C\beta_{N-1A} & C\beta_{2T} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ C\alpha_{1T} & C\alpha_{2T} & \cdots & C\alpha_{N-1T} & C\alpha_{NT} \\ C\beta_{1T} & C\beta_{NA} & \cdots & C\beta_{N-1T} & C\beta_{NT} \end{pmatrix} \begin{pmatrix} Ch_1 \\ Ch_2 \\ \vdots \\ Ch_{N-1} \\ Ch_N \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_{N-1} \\ \varepsilon_N \end{pmatrix} = \begin{pmatrix} \alpha_A \\ \beta_A \\ \vdots \\ \alpha_T \\ \beta_T \end{pmatrix}$$

5. Validation

In this section, we describe the validation method and the result of the proposed approach. To validate the results of the model, we perform the cross-validation because we have little project data to do the test-set validation. Cross-validation is an approach to estimate how well the model we've made from some training data is going to perform on future data. Also, we analyze the magnitude of relative error (MRE) and the mean squared-error (MSE) between the actual values and the predicted values from the proposed model.

5.1. Evaluation Criteria

A good software estimator generates predictions "close to" actual known data. PRED(X) is one such measure of "closeness", which is computed from the magnitude of relative error (MRE) which is the relative size of the difference between the actual and estimated value [19]. PRED(X) reports the average percentage of estimates that were within X% of the actual values. For example, PRED(30) = 64% means that 64% of the estimates are within 30% of the actual.

On the other side, we use the mean squared error (MSE) as another evaluation criterion for validating results. The MSE of an estimate x_{est} of x is the expected value of the squared norm of the estimation error. It is also equal to the sum of the variance and the squared norm of the bias [20].

5.2. Results of Total Defect Distributions

First, we show the MMRE between predicted defect distributions and actual defect distributions to compare prediction accuracy in Table 3. The error values are somewhat fluctuated, but the proposed model using 18 projects yields PRED(30) of 72% in requirement analysis phase, PRED(30) of 83% in design phase, PRED(30) of 94% in coding phase and PRED(30) of 56% in testing phase. Thus, the proposed model yields PRED(30) of 72% for all phases on average.

Table 3. MMRE and PRED(30) by phases

Phases	RA MRE	Des MRE	Cod MRE	Test MRE	MMRE
MMRE	27%	20%	12%	48%	27%
PRED(30)	72%	83%	94%	56%	72%

The MSE between predicted defect distributions and actual defect distributions to compare prediction accuracy is shown in Table 4.

Table 4. MSE by phases

Phases	RA Err^2	Des Err^2	Cod Err^2	Test Err^2	MSE
MSE	0.00198	0.00531	0.00475	0.00118	0.00331

5.3. Results of Defect Type Distributions

The error values are also fluctuated, but the proposed model yields PRED(30) of 67% in requirement analysis phase, PRED(30) of 67% in design phase, PRED(30) of 83% in coding phase and PRED(30) of 83% in testing phase. Thus, the proposed model yields overall PRED(30) of 75% for all phases and all defect types on average.

Table 5. Average PRED(30) by defect types and phases

Defect type	RA	Design	Cod	Test	Ave.
F	44%	78%	94%	89%	76%
P	72%	56%	100%	72%	75%
S	67%	56%	67%	94%	71%
C	50%	78%	50%	33%	53%
E	94%	94%	44%	89%	80%
Average	67%	67%	83%	83%	75%

The MSE between predicted defect distributions and actual defect distributions to compare prediction accuracy is shown in Table 6. The MSE is 0.01002 in requirement analysis phase, 0.00627 in design phase, 0.00288 in coding phase, 0.00515 in testing phase. Thus, the proposed model yields the overall MSE, 0.00608, for all phases and defect types on average.

Table 6. MSE by defect types and phases

Defect type	RA Err^2	Des Err^2	Cod Err^2	Test Err^2	MSE
F	0.00908	0.00506	0.00206	0.01490	0.00778
P	0.00908	0.00506	0.00118	0.00260	0.00448
S	0.01364	0.00966	0.00429	0.00260	0.00755
C	0.01646	0.01072	0.00429	0.00372	0.00880
E	0.00185	0.00084	0.00259	0.00191	0.00180
MSE	0.01002	0.00627	0.00288	0.00515	0.00608

The Figure 8 shows an example of the distributions of actual defect data, estimated data by MLE and predicted data by the proposed model for a project in order to compare the results.

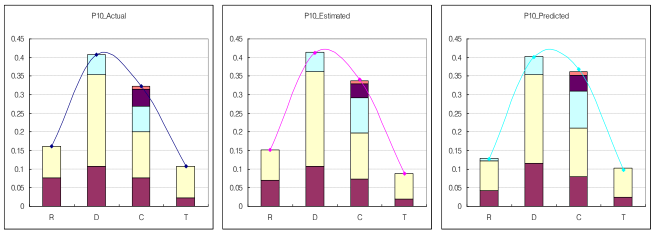


Figure 8. Actual, estimated and predicted defect type dist.

6. RELATED WORKS

Many research papers proposed several defect prediction models such as the empirical phase based defect prediction model [7], Rayleigh model [8][9], the constructive quality model (COQUALMO) [2][3], the artificial neural network based model [10][11] and the Bayesian Belief Network based model [12][13], and so on.

Empirical phase-based prediction model supports phase-by-phase prediction and tracking of the number of defects likely to be encountered during development. It is an extension of the test defect density metric. It means that the model takes a set of defect injection rates and defect removal rates of each development phase as input, and then models the defect removal pattern. With various tools, this predicted information can assist project managers in planning projects and in assessing project progress against expected results at phase ends [7].

Rayleigh model assumes that detecting defects in the different development phases follow a Rayleigh distribution function. It has been empirically well established that software projects follow a lifecycle pattern described by Rayleigh density curve. Using this property, it models the total defect distribution of the entire development phases [9]. To predict defects, it is required to estimate a parameter using the historical time at which the curve reaches its peak. After the parameter is estimated, the shape of the entire curve can be determined, and project managers can predict total defect distribution of the current project to manage.

The COQUALMO [2] is an extension of the existing COCOMO II model to predict the number of defects in the different development phases (requirement, design and coding) [3][10]. This model has two sub-models which are analogous to the 'tank and pipe' model: the Defect Introduction (DI) and Defect Removal (DR) models. The DI model's inputs include source lines of code (SLOC) and/or function points (FP) as the sizing parameter, adjusted for both reuse and breakage and a set of 21 multiplicative DI-drivers divided into four categories; platform, product, personnel and project. These 21 DI-drivers are a subset of the 22 cost parameters required as input for COCOMO II. Using COCOMO II drivers not only makes it relatively straightforward to integrate COQUALMO with COCOMO II but also simplifies the data collection activity which has already been set up for COCOMO II [2][11].

One of the concerns in software reliability research is how to develop general defect prediction models. It is because existing models typically rely on assumptions about development environments, the nature of software defects and the probability of individual defects occur. Recent advances in neural networks show that they can be used in applications that involve predictions that do not require making assumptions about either the development environment or external parameters [11]. The neural network is trained with a series of inputs (size, complexity and other characteristics) and the correct output (defect density or number of defects) from the training data so as to minimize the prediction error. Once the training is complete, and the appropriate weights for the network arcs have been determined, new inputs can be presented to the network to predict the corresponding estimate of the response variable [3].

The Bayesian Belief Networks (BBN) is used to deal with causal relationships among variables that allow uncertainty for some variables [12]. Using BBN, we are able to express complex interrelations within the model at a level of uncertainty commensurate with the problem in defect prediction [13]. It is possible to represent expert beliefs about the dependencies between different variables and propagate consistently the impact of evidence on the probabilities of uncertain outcomes, such as 'future system reliability.' Actually BBN provides prediction of defects with reflecting expert opinions in each individual node.

As mentioned earlier, however, the software defect prediction works have focused on the number of defects in a software system with code metrics, inspection data, and process-quality data. Our concern is that such information cannot be sufficient for project planning and management.

7. Conclusion and Future Works

In this paper, we proposed a new approach for predicting distribution of defects and their types based on project characteristics. For this approach, the model for prediction was established using the curve fitting method and the regression analysis. The maximum likelihood estimation was used in fitting the Weibull probability density function to the actual defect data, and the regression analysis is used to identify the relationship between the project characteristics and the Weibull parameters. In the prediction, the distribution of defects with their types for different phases in the project was estimated based on the past projects data. To do so, we first predicted the entire distribution of defects by all phases, and then we predicted distribution of defect types per each phase of project.

To validate the results of the proposed model, we used the magnitude of relative error (MRE) and the mean squared error (MSE) between the actual values and the predicted values from the proposed model. To sum up, the proposed model yielded PRED(30) of 72% on average for prediction of the entire distribution of defects, and it yielded the overall MSE of 0.00331 on average for all phases. Besides, for all phases and all defect types, the proposed model yields overall PRED(30) of 75% and overall MSE of 0.00608 on average.

From these experiments and the results, we can use this proposed model as a defect prediction model for project planning and management at the early phase. The defect prediction with types of defects can support phase-by-phase forecasting and tracking the trend of detecting such defects likely to be encountered during development. Thus, this information will assist project managers in reasonably planning projects and in assessing project progress against expected results at interim points.

However, like most performance metrics in software, it can provide large variations. To make accurate and stable predictions of defects found in complex software projects, we need a rich set of process factors as input factors of the model. Thus, it is considered to use various characteristics of project like the DI drivers of COQUALMO as input factors. Also, it will help make a robust prediction to apply the orthogonal defect classification (ODC) as classification method of defects. Besides, if the data will be available sufficiently, we will be able to apply case base reasoning (CBR) as a preprocessor to make groups with projects having similar characteristics [21].

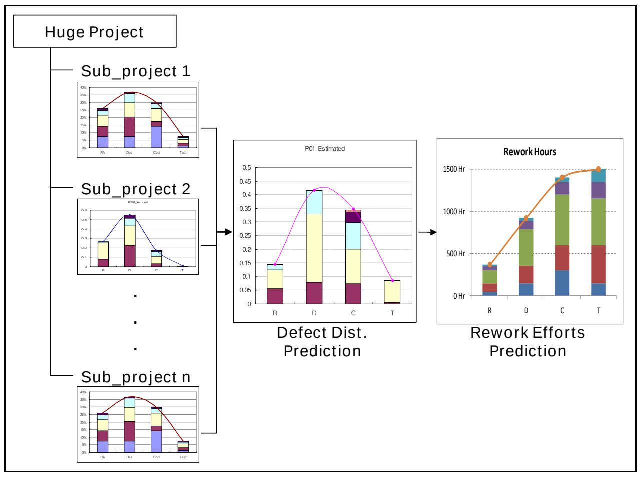


Figure 9. Consolidation of defect and rework efforts prediction

Lastly, when huge scale projects are executed, they can be decomposed to several sub-projects for managerial effectiveness and efficiency. In such cases, the sub-projects can have their own characteristics differently. Thus, their defect distributions can be predicted using our model according to each sub-project's characteristics, and be consolidated to one overall prediction. Through this, the amount of efforts to be reworked will be derived from the prediction of defect distributions. It will be more helpful for the project managers and members to prepare for project planning and management. The Figure 9 shows the concept of the consolidation of each defect prediction and prediction of rework efforts in case of huge projects. For this, historical data of project characteristics and defects in various phases should be absolutely accumulated in organizations.

References

[1] Paulk, M.C., Weber, C.V., "The Capability Maturity Model: Guidelines for Improving the Software Process", *Addison-Wesley*, MA, 1995.

[2] Standards Organization, "Software Product Evaluation-Quality Characteristics and Guidelines for their Use", Chulani S., "COQUALMO (COnstructive QUALity MOdel) a software defect density prediction model," *Proceedings of the ESCOM SCOPE'99*, 297-306, 1999.

[3] Chulani S., "Bayesian Analysis of Software Cost and Quality Models", *University of Southern California*, 1999.

[4] Qinbao Song, Martin Shepperd, "Software Defect Association Mining and Defect Correction Effort Prediction," *IEEE Transactions on Software Engineering*, vol. 32, no. 2, pp. 69-82, Feb., 2006.

[5] P. Jalote, S. Raghavan, and S. Ramakrishna, "Quantitative Quality Management through Defect Prediction and Statistical Process Control" *Proc. Second World Quality Congress for Software*, Sept. 2000.

[6] Ch. Ali Asad, Muhammad Irfan Ullah, "An Approach for Software Reliability Model Selection," *COMPSAC 2004*, 534-539, 2004.

[7] Robyn N., Colin C., "Software Process Improvement Journey: IBM Australia Application Management Services", *CMU/SEI-2005-TR-002*, 2005.

[8] Stephen H. Kan, "Metrics and Models in Software Quality Engineering," 2nd ed., *Addison-Wesley*, 2002.

[9] Samuel King, "Progressive Software Reliability Modeling", *ISSRE*, 1999.

[10] Hans S., "Design of a Methodology to Support Software Release Decisions", *Univ. of Groningen*, 2005.

[11] Karunanithi, N. Malaiya, Y. K., "Using neural networks in reliability prediction," *IEEE Software*, 1992.

[12] S. Amasaki, Y. Takagi, O. Mizuno, Tohru Kikuno, "A Bayesian Belief Network for Assessing the Likelihood of Fault Content", *ISSRE*, 2003.

[13] Fenton, N. E. and Neil, M. "A Critique of Software Defect Prediction Models," *IEEE Transactions on Software Engineering*, 25(5), 675-689, 1999.

[14] Jongmoon Baik, "The Effects of Case Tools On Software Development Effort", PhD Thesis, *University of Southern California*, 2000.

[15] T. T. Mattila, J. K. Kivilahti, "Metallurgical factors behind the reliability of high-density lead-free interconnections", *Springer Science+Business*, 2005.

[16] Potts, W., "Survival Data Mining Predictive Hazard Modeling for Customer History Data", *SAS Institute*, 2003.

[17] Maximum likelihood estimation, Wikipedia http://en.wikipedia.org/wiki/Maximum_likelihood_estimation

[18] Fenton N. E, Neil M, Marsh W, "Project Data Incorporating Qualitative Factors for Improved Software Defect Prediction", *ICSE, PROMISE workshop*, 2007.

[19] Tim Menzies, Dan Port, Zhihao Chen, "Simple software cost analysis: safe or unsafe?", *ACM SIGSOFT Software Engineering Notes*, v.30 n.4, July 2005.

[20] Eldar,Y.C., Ben-Tal, A., "Robust mean-squared error estimation in the presence of model. Uncertainties", *IEEE Trans. Signal Processing*, vol. 53, Jan. 2005.

[21] Khoshgoftarr et al., "Predicting Fault-Prone Modules with Case-Based Reasoning", *Proceedings of the 8th ISSRE*, Nov. 1997.