A Value-Added Predictive Defect Type Distribution Model based on Project Characteristics

Youngki Hong, Jongmoon Baik, In-Young Ko, Ho-Jin Choi
School of Engineering
Information and Communications University
119 Munji-ro, Yuseong-Gu, Daejeon
Republic of Korea
{young}@SKCC.COM , {jbaik, iko, hjchoi}@icu.ac.kr

Abstract

In software project management, there are three major factors to predict and control; size, effort, and quality. Much software engineering work has focused on these. When it comes to software quality, there are various possible quality characteristics of software, but in practice, quality management frequently revolves around defects, and delivered defect density has become the current de facto industry standard. Thus, research related to software quality has been focused on modeling residual defects in software in order to estimate software reliability. Currently, software engineering literature still does not have a complete defect prediction for a software product although much work has been performed to predict software quality.

On the other side, the number of defects alone cannot be sufficient information to provide the 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 about software quality such as in-process defects, their types, and so on. In this paper, we propose a new approach for predicting the distribution of defects and their types based on project characteristics in the early phase. For this approach, the model for prediction was established using the curve-fitting method and regression analysis. The maximum likelihood estimation (MLE) was used in fitting the Weibull probability density function to the actual defect data, and regression analysis was used in identifying the relationship between the project characteristics and the Weibull parameters. The research model was validated by cross-validation.

KEY WORDS

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

1. Introduction

For producing reliable software products and managing projects visibly, 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 [1]. To represent quality of products, there are various possible quality characteristics of software, and there is even an international standard for this. In practice, 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 [2][3]. 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 [4]. 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.

In this paper, we address the problems mentioned above by proposing a defect type distribution prediction with project characteristics information. This approach can support to plan suitable quality assurance activities and prevent possible defects. It can also help us to reduce the efforts of performing reworks and the cost of producing high quality software.

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

978-0-7695-3131-1/08 $25.00 © 2008 IEEE
DOI 10.1109/ICIS.2008.36

presents the validation of our proposed approach. Lastly, the conclusion and future research directions are drawn.

2. Related Works

Many researches proposed several defect prediction models such as Rayleigh model [5][6], the constructive quality model (COQUALMO) [1][7], the empirical phase based defect prediction model [3][5], and the Bayesian Belief Network based model [8][9], and so on. We briefly discuss major methods to use in predicting defects.

The Rayleigh model assumes that detecting defects in the different development phases will follow a Rayleigh distribution function which is a special case of Weibull distribution function. It has been empirically well established that software projects follow a lifecycle pattern described by the Rayleigh density curve. Using this property, it models the total defect distribution of the entire development phases [6]. To predict defects, it is required to estimate a parameter using the historical time at which the curve reaches its peak. The area below the curve up to the peak is about 40% of the total area [5]. 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 [1] is an extension of the existing COCOMO II model to predict the number of defects in the different development phases [7][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 and/or function points 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 [1].

The empirical phase-based defect 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. Supported by various tools, this information assists project managers in accurately planning projects and in assessing project progress against expected results at interim points [5].

The Bayesian Belief Networks (BBN) is used to deal with causal relationships among variables that allow uncertainty for some variables [8]. There are positive forward-looking approaches that model the complexities of software development using new probabilistic techniques. Using BBN, we are able to express complex interrelations within the model at a level of uncertainty commensurate with the problem in defect prediction [9]. 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 reflects expert opinions in each individual node. In other words, BBN models the subjectivity and uncertainty that is pervasive in software development.

3. Research Approach

3.1. Overview of the Proposed Model

As mentioned earlier, the researches of software defect prediction 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.

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.

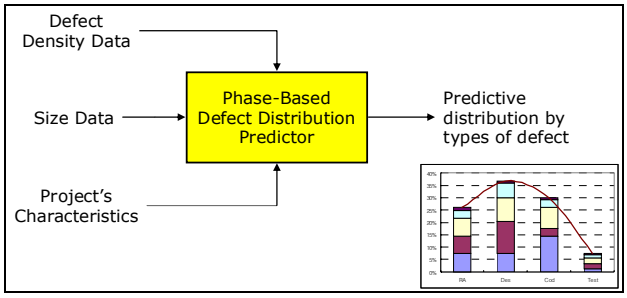

Figure 1. Overview of the proposed model

3.2. Approach

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

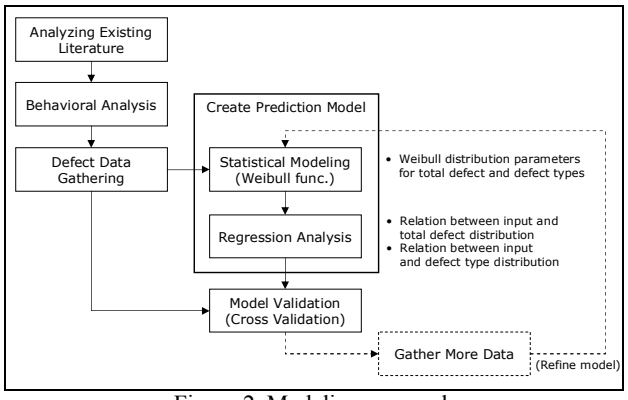

Figure 2. Modeling approach

Step 1) Analyzing existing literature : 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 [11].

Step 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. This can show the behavioral effects of higher vs. lower levels of each factor on project quality levels [11].

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

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

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

Step 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.

Step 7) Gathering of more data for refining the model : We can continue to gather data and refine the model to be more reliable.

3.3. Weibull Distribution Function

The Weibull distribution is defined by the following probability density function.

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

The purpose of using the Weibull function is to characterize the failure distribution and to make inferences about the failure mode. It can be used with relatively small sample sizes. 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 that is widely used in engineering applications. The parameter estimates can be used to identify other characteristics of the distribution [12].

3.4. 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 [13].

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:

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

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

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

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

4. Data Description

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

4.1 Defect Data

In addition to project characteristics, we also used defect data obtained at all phases in the development life cycle (requirement analysis, design, coding, and testing) from the 18 software projects. The defect data were

acquired from requirement review, design review, code inspection, and testing by project members and testers.

The gathered defect data were recorded according to the organizational standard defect types as project members detected them during the phases of the projects. There are five types of defects in the gathered data: consistency, function, standard, performance, and miscellaneous. Table 1 shows brief explanations of defect types.

Table 1. Types of defects

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

The defect type distribution per development phase can be modeled according to the Weibull distribution because they follow the characteristics of the Weibull distribution (convex distribution form and monotonous decrement after a peak value) shown in Figure 3.

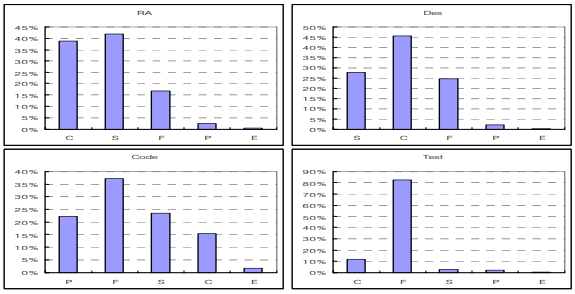

Figure 3. Average defect type distribution in each phase

4.2 Project Characteristics

For this model construction, we gathered data from 18 completed software development projects which were performed in public business sector. There can be lots of quantitative and qualitative characteristics for differentiating software projects in each organization [5] [14]. In our case, the 11 factors of project characteristics are included in three categories: project basic information, human resource information and development 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 2.

Table 2. 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
	Industry area	Industry area that the system is applied in
	Development methodology	Type of development methodology used in the project
	Defect density	Total defect density per each project
Human resource information		
	# of members	Total number of project members
	Project member ratio	Ratio between our own members and total number of project members
	Productivity	Project productivity which is function points divided by number of person months
Development information		
	Development language	Type of development language used in the project
	DBMS type	Type of DBMS used in the project

5. 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.

5.1. 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 (Requirement analysis, Design, Coding, and Test). 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	Curve	α_A	β_A
Prj 1	d_{1R}	d_{1D}	d_{1C}	d_{1T}		α_{1A}	β_{1A}
Prj 2	d_{2R}	d_{2D}	d_{2C}	d_{2T}		α_{2A}	β_{2A}
...	...	...	...	...	...	...	...
Prj M	d_{MR}	d_{MD}	d_{MC}	d_{MT}		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. 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.

5.2. 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 5 shows the overview of regression analysis between project characteristics and the parameters of Weibull distributions.

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_{MN}

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}

Regression Analysis

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 5. Regression analysis for parameter prediction

5.3. 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.

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

6. Validation

In this section, we describe the validation method and the result of the proposed approach. For validating results, we perform cross-validation using the data which we used to develop the model because of the lack of samples. Also, we analyze the magnitude of relative error (MRE) and the PRED(30) between the actual values and the predicted values from the proposed model.

6.1. 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. 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 [7].

6.2 Results of Total Defect Distributions

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	27%	20%	12%	48%
PRED(30)	72%	83%	94%	56%

6.3. Results of Defect Type Distributions

The error values fluctuate somewhat, but the proposed model yields a PRED(30) from 50% to 94% in the requirement analysis phase, a PRED(30) from 56% to 94% in the design phase, a PRED(30) from 44% to 97% in the coding phase, and a PRED(30) from 33% to 94% in the testing phase. Thus, the proposed model yields an overall PRED(30) of 75% on average for all phases and all defect types.

Table 4. Average PRED(30) by defect types and phases

Defect type	RA	Design	Cod	Test	Ave.
Function	44%	78%	94%	89%	76%
Performance	72%	56%	97%	72%	75%
Standard	67%	56%	67%	94%	71%
Consistency	50%	78%	50%	33%	53%
Misc.	94%	94%	44%	89%	80%

The Figure 6 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.

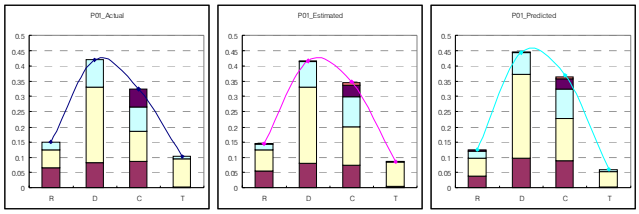

Figure 6. Actual, estimated, and predicted defect type dist.

7. Conclusion and Future Works

We discussed a new approach for predicting the distribution of defects and their types based on project characteristics. For this approach, the maximum likelihood estimation was used in fitting the Weibull probability density function to the actual defect data, and the regression analysis was used to identify the relationship between the project characteristics and the Weibull parameters. In the prediction, the distribution of defects and their types for different phases in the project was estimated based on past projects data. We predicted the entire distribution of defects by all phases, and then we predicted the distribution of defect types for each phase of project. To validate the results of the proposed model, we used the magnitude of relative error between the actual values and the predicted values from the proposed model.

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 aid in phase-by-phase forecasting and monitoring of the trend of defects likely to be encountered during software development. Thus, this information will assist project managers in reasonably planning projects and in assessing project progress against expected results at interim points. For this, historical (application-/project-specific) data should absolutely be accumulated in organizations to predict defect levels.

However, like most performance metrics in software, it can provide large variations. To make accurate and stable predictions of defects found in complex and various software projects, we need a rich set of process factors as input factors of the model. Thus, we need to try to apply the DI-drivers of COQUALMO as input factors to the proposed model. Especially, we have to concentrate on the factor of human resources such as analyst's capability, programmer's capability, experience about similar applications, experience about platform, experience of development languages and supporting tools, personnel continuity, and so on.

Acknowledgment

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Advancement)" (IITA-2008-(C1090-0801-0032))".

References

[1] Chulani S., "COQUALMO (COnstructive QUALity MOdel) a software defect density prediction model," *Proceedings of the ESCOM SCOPE'99*, 297-306, 1999.
[2] 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.
[3] 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.
[4] Ch. Ali Asad, Muhammad Irfan Ullah, "An Approach for Software Reliability Model Selection," *COMPSAC 2004*, 534-539, 2004.
[5] Stephen H. Kan, "Metrics and Models in Software Quality Engineering," 2nd ed., *Addison-Wesley*, 2002.
[6] Samuel King, "Progressive Software Reliability Modeling", *ISSRE*, 1999.
[7] Chulani S., "Bayesian Analysis of Software Cost and Quality Models", *University of Southern California*, 1999.
[8] S. Amasaki, Y. Takagi, O. Mizuno, Tohru Kikuno, "A Bayesian Belief Network for Assessing the Likelihood of Fault Content", *ISSRE*, 2003.
[9] Fenton, N. E. and Neil, M. "A Critique of Software Defect Prediction Models," *IEEE Transactions on Software Engineering*, 25(5), 675-689, 1999.
[10] Hans S., "Design of a Methodology to Support Software Release Decisions", *Univ. of Groningen*, 2005.
[11] Jongmoon Baik, "The Effects of Case Tools On Software Development Effort", PhD Thesis, *University of Southern California*, 2000.
[12] Potts, W., "Survival Data Mining Predictive Hazard Modeling for Customer History Data", *SAS Institute*, 2003.
[13] Maximum likelihood estimation, Wikipedia http://en.wikipedia.org/wiki/Maximum_likelihood_estimation
[14] Fenton N. E, Neil M, Marsh W, "Project Data Incorporating Qualitative Factors for Improved Software Defect Prediction", *ICSE, PROMISE workshop*, 2007.