Bayesian statistical models for predicting software development effort

C. van Koten [1]

Department of Information Science,University of Otago, P.O.Box 56, Dunedin, New Zealand

Abstract

Constructing an accurate effort prediction model is a challenge in Software Engineering. This paper presents new Bayesian statistical models, in order to predict development effort of software systems in the International Software Benchmarking Standards Group (ISBSG) dataset. The first model is a Bayesian linear regression (BR) model and the second model is a Bayesian multivariate normal distribution (BMVN) model. Both models are calibrated using subsets randomly sampled from the dataset. The models' predictive accuracy is evaluated using other subsets, which consist of only the cases unknown to the models. The predictive accuracy is measured in terms of the absolute residuals and magnitude of relative error. They are compared with the corresponding linear regression models. The results show that the Bayesian models have predictive accuracy equivalent to the linear regression models, in general. However, the advantage of the Bayesian statistical models is that they do not require a calibration subset as large as the regression counterpart. In the case of the ISBSG dataset it is confirmed that the predictive accuracy of the Bayesian statistical models, in particular the BMVN model is significantly better than the linear regression model, when the calibration subset consists of only five or smaller number of software systems. This finding justifies the use of Bayesian statistical models in software effort prediction, in particular, when the system of interest has only a very small amount of historical data.

Key words: Effort prediction, Bayesian statistics, Regression, Software metrics

[1] Corresponding author. Tel.: +64-3-479-8142; fax: +64-3-479-8311.
E-mail address: ckoten@infoscience.otago.ac.nz

1 Introduction

Accurate effort prediction at an early stage is often an important factor for successful software development. Hence software development effort prediction models are useful tools for software practitioners and of active interest of researchers. The related studies to date have proposed a number of models, some of which are generic across different software systems and the others specific for a particular type of software systems and/or a development environment. However, unfortunately it can be said that constructing an accurate effort prediction model for a software system, in general, still remains as one of the greatest challenges in Software Engineering.

Software development effort can vary depending on a large number of factors, some known and the others often unknown. Consequently the influential factors and their predictive relationship in one software system can be very different from another system. Hence, in order for a software development effort prediction model to be accurate, the model usually needs to be calibrated using historical data collected from a number of software systems, which are considered to be similar to the system of interest. However, unfortunately it is often the case that the amount of such data is very small, even if any. Consequently the small amount of calibration data often limits predictive accuracy of the existing effort prediction models.

This paper proposes new models, which can predict software development effort even if only a very small amount of historical data is available. The models are Bayesian statistical models based on Bayesian inference [5]. Bayesian inference allows a predictive model to be constructed based on the prior subjective knowledge of human experts, and to be calibrated using empirical data. Since Bayesian inference makes use of the expert's prior knowledge for the model construction, Bayesian statistical models based on good prior knowledge can be expected to achieve good predictive accuracy even if only a very small amount of empirical data is available for calibration. In the case of software development effort prediction, it is indeed often the case that software development organizations and practitioners posses good prior knowledge about the system to be developed, through experiences in the past development. Hence it seems natural if the knowledge can be integrated into the model together with available empirical data.

The predictive accuracy of the new Bayesian statistical models are validated using the International Software Benchmarking Standards Group (ISBSG) dataset. The models' predictive accuracy is measured using the absolute residuals and magnitude of relative error. They are also compared with the predictive accuracy of the corresponding linear regression models, since regression models have been popular and often successful for predicting software devel-

opment effort in the previous studies. The term *predictive accuracy* in this paper means how well a predictive model constructed from known cases can predict the outcome of an unknown case. Hence the models are calibrated using subsets randomly sampled from the dataset and validated using different subsets, which consist of only data unknown to the models.

An application of Bayesian statistics to Software Engineering to date is limited to a relatively small number of studies, which used Bayesian data analysis [3,4,15] or Bayesian network [2,1,7,8,16–18,20–22]. Bayesian data analysis is based on the Bayesian probability theory but not on Bayesian inference. Hence the studies are different from this study. Bayesian network is a probabilistic network that uses Bayesian inference [12]. Hence Bayesian network models are a Bayesian statistical model. However, the existing Bayesian network software effort prediction models have a restriction on the use of continuous probability density functions. This restriction consequently limits the models' applicability only to some cases in software effort prediction, where all the variables involved are discrete. On the other hand, the Bayesian statistical models proposed in this paper have no such restriction. Hence the models are applicable to not only the cases that involve discrete variables only, but also the cases that involve continuous variables only, as well as to the cases that involve both discrete and continuous variables.

The structure of the reminder of this paper is as follows. Section 2 introduces the ISBSG dataset, which is used in this study. Section 3 briefly explains Bayesian inference, which forms the theoretical background of Bayesian statistical models. Section 4 describes the proposed Bayesian statistical models, with specific reference to the ISBSG dataset. Section 5 provides the definition of predictive accuracy measures used in this study. This is followed by Section 6, which explains the models' validation method. Section 7 discusses the Bayesian statistical models' predictive accuracy and compares them with the corresponding linear regression models. Finally Section 8 presents conclusions and a direction of future studies.

2 The ISBSG dataset

2.1 Description

ISBSG is an Australian-based business organization who provides a wide range of services for software practitioners. One of their services is building an international data repository of software development/management projects. The data in this repository is currently accessible for academic use under a specific license agreement with the organization. The data used in this study is found

on the ISBSG Repository Data CD Release 8. The CD contains data collected from 2027 software development projects, which were undertaken by various commercial organizations in 13 countries around the world, with the greatest number of them from Australia, Japan, the Netherlands, the UK, and the USA. Over 90% of the projects were completed in 1996 or later. It is a very comprehensive data repository but disclosure of raw data is not permitted due to the terms of the license agreement.

The data used in this study is a subset of the original 2027 projects. For convenience we refer to this subset as the ISBSG dataset in the remainder of this paper. The ISBSG dataset consists of data form 172 projects, which were chosen from the original 2027 projects by satisfying the following conditions:

- The quality of data being rated at the highest by the ISBSG quality reviewers.
- Using a Function Points support tool that follows the IFPUG standard.
- Counting either all the tables of referenced code as a single file or each table as an individual file.
- Effort recoring method being known.
- Being either a new development or enhancement of the existing system.
- Being undertaken by Telecommunication organizations.
- Type of systems developed being a transaction processing system (TPS), management information system (MI) or administration system (AS).
- Containing no missing data.
- Including no extreme values.

The above conditions are applied in order to obtain a reasonably homogeneous subset that contains as many projects as possible. The homogeneity of data is, in general, required by predictive models for achieving good predictive accuracy. In addition, the homogeneity of data can often reduce the number of predictor variables in the models.

2.2 Variables in the dataset

In order to predict development effort of software systems in the ISBSG dataset, five fields are chosen from the 67 fields in the original dataset. They are four categorical fields and one numerical field. The categorical fields are:

(1) Reference Table Approach (TABLE)
(2) Development Type (DTYPE)
(3) Language Type (4GL)
(4) Application Type (TPS)

The numerical field is Function Points (FP). Those fields are candidate predictor variables for the models described in Section 4. The TABLE variable indicates how the tables of referenced code are counted. It has two levels, either all the tables are counted as a single file or each table is counted as an individual file. The DTYPE variable indicates the type of development. It has two levels, either it is a new development or an enhancement of the existing system. The 4GL variable indicates the type of language used for development. It has two levels, either it is a fourth-generation language (4GL) or not. The TPS variable indicates the type of application being developed. It has two levels, either it is a TPS or not. The FP variable is an adjusted Function Points (FP) count. The term *adjusted* here means that the original FP count was adjusted by the project data submitter due to various reasons, and the adjustment was approved by the ISBSG quality reviewers. Those five variables are chosen based on the author's prior subjective knowledge, that they can potentially be influential on software development effort.

Using the above candidate predictor variables, development effort of each individual software system in the ISBSG dataset is predicted. Then the predicted value is compared against the actual value, which is the value of the Normalized Work Effort field in the original dataset. It is called *normalized* since some of the values in this field were recalculated from the initial value, in order to obtain effort equivalent to a full development life-cycle. This recalculation was required only for the projects that did not cover a full development life-cycle. The ISBSG dataset contains no such project. Hence the value of the Normalized Work Effort field in the ISBSG dataset is the actual total effort spent during the project. We refer to the Normalized Work Effort simply as EFFORT in the remainder of this paper. EFFORT were measured in hours. The descriptions of the variables are found in Table 1.

Variable	*Definition*
TABLE	The referenced code tables are counted as a whole or individually
DTYPE	A new development or an enhancement of the existing system
4GL	Fourth-generation language is used or not
TPS	A transaction processing system or not
FP	An adjusted Function Points count
EFFORT	Total effort during the project (hours)

Table 1
Variables in the ISBSG dataset

3 Bayesian inference

3.1 Joint probability

In the probability theory, the *joint probability* distribution of a set of random variables $\mathbf{X}$ is defined as:

$$P(\theta, \mathbf{X}) = P(\mathbf{X} \mid \theta)P(\theta) \tag{1}$$

where θ denotes a set of parameters that describe the *joint probability* distribution. The set of random variables $\mathbf{X}$ can include both predictor variables and the variable(s) to be predicted. In Equation 1, $P(\theta)$ is called the *prior* probability distribution of θ, and $P(\mathbf{X} \mid \theta)$ is called the *likelihood* of data. The *likelihood* of data is the amount proportional to the probability of observing a given set of data, which is a set of specific values of $\mathbf{X}$. The *likelihood* of data is calculated for the given *prior* probability distribution. Hence, Equation 1 shows that the *joint probability* distribution of $\mathbf{X}$ can be found by multiplying the *prior* probability distribution of θ by the *likelihood* of data, provided that the *prior* probability distribution of θ is known. If the *joint probability* distribution of $\mathbf{X}$ is found, the *marginal* probability distribution of an individual variable in the set $\mathbf{X}$ is also found, by integrating the *joint probability* distribution function by the other variables in the set. For example, the *marginal* probability distribution of X_1 in the set $\mathbf{X} : X_1, \cdots, X_n$ is found by integrating the *joint probability* distribution function by $X_2, \cdots, X_n$. The term *marginal* means 'unconditional'. That is, the *marginal* probability of X_i is the probability of a value of X_i being observed regardless of the values of any other variables in the set $\mathbf{X}$.

3.2 Conditional probability and Bayes' theorem

On the other hand, in the probability theory, a relationship of two events X and Y is defined in the *conditional probability*, $P(Y \mid X)$, which is the probability of event Y conditional on a given outcome of event X. The *conditional probability* is calculated using Bayes' Theorem:

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)} \tag{2}$$

where $P(X \mid Y)$ is the *conditional probability* of event X given event Y, and $P(X)$ and $P(Y)$ are the *marginal* probability of events X and Y respectively.

In the Bayes' Theorem, $P(Y)$ and $P(Y \mid X)$ are also called the *prior* probability distribution and the *posterior* probability distribution of a random variable Y, respectively. This is because in the Bayes' Theorem, $P(Y)$, the *prior* probability distribution of Y, is updated to $P(Y \mid X)$, the *posterior* probability distribution of Y, using the given information of $P(X)$ and $P(X \mid Y)$.

Using the Bayes' Theorem 2, it is shown that:

$$P(\theta \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid \theta)P(\theta)}{P(\mathbf{X})} \tag{3}$$

Considering that $P(\mathbf{X})$ is a constant for a given set of specific values of $\mathbf{X}$, the above Equation 3 means

$$P(\theta \mid \mathbf{X}) \propto P(\mathbf{X} \mid \theta)P(\theta) \tag{4}$$

From Equation 1, this means

$$\begin{aligned} &\textit{Posterior distribution of } \theta \ \propto \\ &\textit{Likelihood of data} \ \times \ \textit{Prior distribution of } \theta \end{aligned} \tag{5}$$

Similarly,

$$P(\theta \mid \mathbf{X}) \propto P(\theta, \mathbf{X}) \tag{6}$$

That is,

$$\begin{aligned} &\textit{Posterior distribution of } \theta \ \propto \\ &\textit{Joint probability distribution of } \mathbf{X} \end{aligned} \tag{7}$$

3.3 Prediction

The above relationships 5 and 7 provide the principle of Bayesian inference. In Bayesian inference, the *prior joint probability* distribution of a set of random variables $\mathbf{X}$ is specified and updated using the likelihood of the observed data. Then the *posterior marginal* probability distribution of the variable of interest in the set $\mathbf{X}$ is obtained by integrating the *posterior joint probability* distribution of $\mathbf{X}$ by the other variables in the set. For prediction, both predictor variables and the variable(s) to be predicted are in $\mathbf{X}$. Let us assume

that $\mathbf{X}$ consists of multiple predictor variables: $X_1, \cdots, X_n$ and a single predicted variable, Y. Then, the *posterior marginal* probability distribution of Y is obtained by integrating the *posterior joint probability* distribution of $\mathbf{X}$ by $X_1, \cdots, X_n$.

When the *prior joint probability* distribution of $\mathbf{X}$ is specified based on the subjective knowledge of human experts, Bayesian inference provides a framework in which human experts' knowledge can be incorporated into a predictive model together with empirical data. Bayesian statistical models are based on Bayesian inference. Hence Bayesian statistical models allow the subjective knowledge of human experts, which is expressed as the *prior joint probability* distribution, to be updated using empirical data. In addition, a Bayesian statistical model can be re-calibrated every time additional data become available. Another characteristic of Bayesian statistical models is that they provide an interval estimate of the variable of interest, since Bayesian inference outputs its entire *posterior marginal* probability distribution. This would be an advantage when an interval estimate is preferred to a point estimate. In the case of software development effort prediction, an interval estimate would be particularly useful for risk management. When a point estimate is required, an appropriate summary statistic of the probability distribution should be used, according to the need of the estimator. This study uses the mode statistic as the predicted value, since the mode is the value of EFFORT having the highest probability. In other words, the mode is the EFFORT value that most likely occur. In the case of the Bayesian statistical models proposed in this paper, the mode coincides with the mean of the distribution, which is usually more convenient to calculate.

4 Bayesian statistical models

4.1 Preliminary data analysis

Bayesian statistical models are based on Bayesian inference. Hence, it models the *prior joint probability* distribution of a set of random variables $\mathbf{X}$. For prediction, the set $\mathbf{X}$ includes predictor variables and the variable(s) to be predicted. Hence in this study, $\mathbf{X}$ would consist of the five candidate predictor variables and EFFORT. However, in reality, the influence of some of the candidate predictor variables on EFFORT may not necessarily be statistically significant. Including one or more non-influential predictor variables into a predictive model often leads to a larger standard error of the predicted value. In addition, omitting those non-influential predictor variables makes the model simpler. A simpler model is preferable since it is easier to understand and it requires lesser effort for data collection. Hence, the influence of each individual

Variable	*F statistic*	*D.f.*	*P-value*
LogFP*TABLE	0.823	1 and 162	0.366
LogFP*DTYPE	3.269	1 and 162	0.072
LogFP*4GL	0.087	1 and 162	0.768
LogFP*TPS	3.034	1 and 162	0.083
TABLE	0.007	1 and 166	0.933
DTYPE	0.470	1 and 166	0.494
4GL	0.699	1 and 166	0.404
TPS	3.649	1 and 166	0.058
LogFP	84.634	1 and 170	0.000

Table 2
Influence on LogEFFORT

candidate predictor variable is tested using a procedure called F-test on the *extra sum of squares*. This test is carried out by comparing the *extra sum of squares* explained by one particular predictor variable against the appropriate residual sum of squares. The *extra sum of squares* is the sum of squares left over after subtracting the contributions of all the other predictor variables in the model. The test assumes the distribution of data to be approximately normal. Hence, the FP and EFFORT values are log-transformed to the LogFP and LogEFFORT values respectively.

The results of the above tests are shown in Table 2. Table 2 contains all the individual candidate predictor variables and some two-way interactions between them. Those interactions are included since they may also be influential on LogEFFORT. Using the 5% level of significance, the p-values in Table 2 indicate that only LogFP is significantly influential on LogEFFORT. This suggests that the model consisting of only LogFP and LogEFFORT seems most appropriate. A series of further F-tests are carried out for confirmation. In those tests, the F statistic is tested consecutively by removing one non-influential variable after another, starting from the full model that contains all the variables. This procedure ensures the final model contains only influential predictor variables, since it tests for the influence of not only each individual variable but also their potential correlation. The final model from this procedure coincides with the model previously suggested. Hence, the Bayesian statistical models in this study use only LogFP to predict LogEFFORT. The log-transformations, instead of the original variables, are used in the models. This is because the Bayesian statistical models proposed in this paper also assume normality of data to the extent similar to the corresponding linear regression model.

However, in Table 2 it can also be argued that the non-influential variables still have some level of influence, some of which are close to the 5% level of significance. Hence it would be better if these influences are not completely ignored. These influences are specifically taken into account in the second half of this study, by validating the above models using small clusters of software systems in the ISBSG dataset, in contrast with the first half that uses the whole dataset. The clusters are formed by only the systems that have exactly the same non-influential variable values. In this way, it is expected that all the systems within the same cluster are influenced by the non-influential variables in the same way. Hence validating the models using each cluster independently from the others should remove the influences of the non-influential variables on LogEFFORT.

4.2 Model description

Two different Bayesian statistical models are constructed in this study. The first model is a Bayesian linear regression (BR) model and the second is a Bayesian multivariate normal distribution (BMVN) model. The BR model is based on Bayesian linear regression [5]. A Bayesian linear regression model, in general, has a parametric form similar to the linear regression counterpart. However, it is different since the Bayesian model can specify the *prior joint probability* distribution of the variables, which includes the *prior marginal* probability distributions of the linear regression coefficient parameters. The BR model is specified as follows:

$$\mathsf{LogEFFORT} \sim N(\mu, \tau) \tag{8}$$

$$\begin{aligned}\mu = {} & \beta_0 + \beta_1(X_1 - \bar{X_1}) + \beta_2(X_2 - \bar{X_2}) \\ & + \cdots + \beta_k(X_k - \bar{X_k})\end{aligned} \tag{9}$$

$$\beta_0 \sim N(0, 0.0001) \tag{10}$$
$$\beta_j \sim N(0, 0.001) \quad for \quad j = 1, \cdots, k \tag{11}$$
$$\tau \sim Gamma(0.001, 0.001) \tag{12}$$

The notation 8 means that LogEFFORT has a normal distribution with mean μ and variance $1/\tau$. In Equation 9 there are k predictor variables. For software systems in the ISBSG dataset, $k = 1$, and the chosen predictor variable is LogFP. The notations 10, 11 and 12 specify the *prior marginal* probability distributions of the specific parameters in the model. These probability distributions are also updated to the corresponding *posterior* distributions using

data in Bayesian inference. The notation 10 means that the linear regression coefficient parameter β_0 has a normal distribution initially with mean 0 and variance $1/0.0001 = 10000$. The initial mean is 0 because prior to calibration, it is equally possible for this coefficient parameter to take either a positive or negative value. On the other hand, for the initial variance, any number relatively larger than the anticipated β_0 would be sufficient. The value of 10000 is chosen in this study. Similarly the notation 11 means that each of the remaining linear regression coefficient parameter β_j has a normal distribution initially with mean 0 and variance $1/0.001 = 1000$. The initial mean is 0 from the same reason as was mentioned above. Similarly the initial variance of 1000 is chosen, although any relatively large number would be sufficient. The notation 12 means that τ, which is the reciprocal of the variance of the *prior* distribution of LogEFFORT, has a gamma distribution with scale parameter $1/0.001 = 1000$ and shape parameter 0.001. These parameter values are often recommended by experienced researchers of Bayesian statistics [5].

The second Bayesian statistical model in this paper, the BMVN model uses a multivariate normal distribution, which is a normal distribution defined under multiple variables. The mean of the multivariate normal distribution is the set of the means of those multiple variables, while the variance is the variance-covariance matrix of the variables. The BMVN model's specifications are as follows:

$$\begin{aligned} &\mathsf{LogEFFORT}, X_1, \cdots, X_k \ \sim \ MVN\left(\mu_j, \tau_{jm}\right) \\ &\quad for \quad j = 1, \cdots, k+1 \\ &\qquad\quad m = 1, \cdots, k+1 \end{aligned} \tag{13}$$

$$\mu_j \sim N\left(0, 0.000001\right) \tag{14}$$

$$\tau_{jm} \sim Wishart\left(C_{jm}, k+1\right)$$

$$where \quad C_{jm} = \begin{cases} 1 & \text{if } j = m \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

The specification 13 means that LogEFFORT and k predictor variables $X_1, \cdots, X_k$ together form a multivariate normal distribution with $mean = \mu_1, \cdots, \mu_{k+1}$ and $variance =$:

$$\begin{pmatrix} 1/\tau_{1,1} & \cdots & 1/\tau_{1,k+1} \\ \vdots & \ddots & \vdots \\ 1/\tau_{k+1,1} & \cdots & 1/\tau_{k+1,k+1} \end{pmatrix}$$

For software systems in the ISBSG dataset, $k = 1$, μ_1 = the mean of LogEFFORT, and μ_2 = the mean of LogFP. The notations 14 and 15 specify the *prior marginal* probability distributions of two specific parameters in the model. The notation 14 means that each mean μ_j has a normal distribution initially with mean 0 and variance $1/0.000001 = 1000000$. The initial mean is chosen to be 0, assuming that prior to calibration there is absolutely no knowledge about the mean values of the variables. The initial variance is chosen to be 1000000, although any number larger than the anticipated variances of LogFP and LogEFFORT would be sufficient. The notation 15 means that τ_{jm} has a Wishart distribution whose initial scale matrix is a $(k+1) \times (k+1)$ identity matrix.

4.3 Model construction

The above BR and BMVN models are constructed using a software tool called WinBUGS, the Windows version of BUGS (Bayesian analysis Using Gibbs Sampling). A version of WinBUGS can be obtained from Imperial College and Medical Research Council in the U.K. under the license agreement. This study used the version 1.4.1. WinBUGS is an implementation of a numerical method of Bayesian inference. This tool allows users to build a complex Bayesian statistical model using a wide range of known distribution functions, including the distributions appeared in the models' specifications above. Then Bayesian inference is made by approximating the *posterior marginal* probability distribution of the variable(s) of interest using a large number of random samples drawn from the *posterior joint probability* distribution by the Gibbs sampler. The Gibbs sampler is a Markov chain Monte Carlo (MCMC) algorithm [11].

For the BR and BMVN models, the effort estimation procedure works as follows. The Gibbs sampler successively samples a value from the *posterior marginal* probability distribution of LogEFFORT, given the value of LogFP. After a number of iterations, this sampling process is known to settle into a dynamic equilibrium, in which the sampling distribution is exactly proportional to the *posterior marginal* probability distribution. In other words, the sampling distribution of LogEFFORT can approximate the *posterior marginal* probability distribution. Hence, once the equilibrium is reached, the predicted LogEFFORT value is obtained by calculating the mean of the sampling distribution. The mean is used because our LogEFFORT estimate, the mode of the sampling distribution, coincides with the mean for the BR and BMVN models.

The number of iterations required to reach the dynamic equilibrium generally depends on the complexity of the model and the initial values chosen for

the sampling algorithm to start with. Hence setting appropriate initial values often reduces the number of iterations. This study set the initial value of τ to 1 in the BR model. This study also set the initial values of τ_{jm} in the BMVN model equal to those of a 2×2 identity matrix. The initial values of μ_j in the BMVN model were also set to the corresponding mean values, which were calculated from the calibration data. As the result, it was observed that both BR and BMVN models reached the equilibrium almost instantly. However, in order to ensure the equilibrium is reached, this study discarded the first 75000 samples and used only the following 75000 samples to approximate the *posterior marginal* probability distribution.

4.4 Model calibration

The whole ISBSG dataset

The BR and BMVN models are first calibrated using the whole ISBSG dataset. The whole dataset here actually means a calibration subset consisting of 115 software systems that were randomly sampled from the ISBSG dataset. The remaining 57 systems form a validation dataset, which is used in Section 7 for model evaluation later in this paper. For convenience we refer to this calibration subset as Calibration-W in the reminder of this paper. Splitting a dataset into two subsets, one for calibration and the other for validation, is a common practice for validating a predictive model. The splitting can be done in many different ways, producing subsets consisting of a different proportion of the cases in the original dataset. The calibration subset here consists of approximately 2/3 of the cases in the ISBSG dataset and the validation subset, the remaining 1/3. It can be argued that the size of Calibration-W is rather small in comparison with the original ISBSG dataset. However, the calibration subset consisting of 115 cases is sufficient for this study, since the interest of this study is to examine the BR and BMVN models' predictive accuracy when a very small calibration subset is used. That is, the role of Calibration-W is to provide the models' predictive accuracy when the calibration subset is not so small, for comparison. Hence how many cases Calibration-W contains is not a concern as long as it is not very small.

Secondly, the models are calibrated using four very small subsets that consist of only five software systems each. These subsets were also randomly sampled from the ISBSG dataset. We refer to those four small calibration subsets as Calibration-W-S1 $\sim$ -S4 for convenience. For Calibration-W-S1 $\sim$ -S4, each remaining 167 systems form the corresponding validation dataset. In order to reduce the possibility of getting a result by chance, it could be argued that taking more than four samples would be better here. However, validating a model using 167 cases is very time consuming. In addition, the possibility of

getting a result by chance is considerably reduced by taking four samples in comparison with taking only one sample, and a total of $167 \times 4 = 668$ cases are already in the validation subsets. Hence this study considers these four samples here would be sufficient.

Clusters of systems

In the second half of this study, the BR and BMVN models are calibrated using clusters of software systems, which were selected from the ISBSG dataset by having exactly the same values for the four non-influential predictor variables:TABLE, DTYPE, 4GL and TPS. As was explained in Section 4, these clusters are specifically used in order to remove the non-significant, however, still existing influences of the non-influential variables on LogEFFORT. The ISBSG dataset contains 16 such clusters. However, four of those clusters consist of only two or less software systems. Hence those clusters cannot be used. In addition, there are five other clusters, whose linear regression effect is statistically not significant, as indicated by the non-significant t-statistic value of the coefficient parameter β_1. When the linear regression effect is not significant, the validity of the linear model is questionable. Hence, those five clusters are not used in this study, either. This results in using the remaining seven clusters:

Cluster 1: 12 enhancement systems, which are not a TPS and did not use 4GL, and whose reference code tables were counted individually.
Cluster 2: 27 enhancement systems, which are not a TPS but used 4GL, and whose reference code tables were counted as a single file.
Cluster 3: 17 enhancement systems, which are not a TPS but used 4GL, and whose reference code tables were counted individually.
Cluster 4: 24 enhancement systems, which are a TPS and used 4GL, and whose reference code tables were counted as a single file.
Cluster 5: 12 newly developed systems, which are a TPS but did not use 4GL, and whose reference code tables were counted as a single file.
Cluster 6: 37 enhancement systems, which are a TPS but did not use 4GL, and whose reference code tables were counted as a single file.
Cluster 7: 10 enhancement systems, which are a TPS but did not use 4GL, and whose reference code tables were counted individually.

First, a leave-one-out validation is performed using each of the seven clusters. The leave-one-out validation means that the models are calibrated using all the software systems except one in one cluster. Then LogEFFORT of the one system that was not used in the calibration, is predicted by the models. This procedure is repeated until LogEFFORT of all the systems in the cluster are predicted. For example, in the case of Cluster 1, the models are calibrated using 11 systems among the 12 systems, and LogEFFORT of the remaining one system is predicted by the models. This procedure is repeated 12 times

to predict LogEFFORT of all the 12 systems in the cluster. We refer to those 12 calibration subsets simply as Calibration-C-1 in the remainder of this paper. The leave-one-out validation is performed independently in each cluster, resulting in Calibration-C-1 $\sim$ -7 respectively. The leave-one-out validation method is chosen since each cluster is relatively small in comparison with the whole dataset. The leave-one-out method allows the largest possible subset in each cluster to be used for calibration.

Secondly, similar to the whole dataset, in order to examine the models' predictive accuracy when the calibration dataset is very small, the models are calibrated using only three software systems, which were randomly sampled from the same cluster. The remaining systems form a validation dataset. A different number of such calibration samples were drawn independently from each cluster, since each cluster consists of a different number of systems. In addition, since it is very time consuming to repeat this procedure for all the seven clusters, only three clusters were used here. They are Clusters 1, 4 and 7. Those clusters were randomly chosen from the seven clusters. From Cluster 1, 18 independent calibration samples were drawn. We refer to those samples as Calibration-C-1-S1 $\sim$ -S18. Similarly, 8 and 23 independent samples were drawn from Clusters 4 and 7, and referred to as Calibration-C-4-S1 $\sim$ -S8 and Calibration-C-7-S1 $\sim$ -S23 respectively.

5 Predictive accuracy measures

This study evaluates and compares the BR and BMVN models using the following predictive accuracy measures: absolute residual (Ab.Res.), the magnitude of relative error (MRE) and pred. Both models are also compared with the corresponding linear regression model using the same measures.

The Ab.Res. is the absolute value of residual given by:

$$Ab.Res. = \mid actual\ value - predicted\ value \mid \tag{16}$$

In this study, the sum of the absolute residuals (Sum Ab.Res.), the median of the absolute residuals (Med.Ab.Res.) and the standard deviation of the absolute residuals (SD Ab.Res.) are used. The Sum Ab.Res. measures the total residuals over the dataset. The Med.Ab.Res. measures the central tendency of the residual distribution. The Med.Ab.Res. is chosen as a measure of the central tendency since the residual distribution is usually skewed in software datasets. The SD Ab.Res. measures the dispersion of the residual distribution.

MRE is a normalized measure of the discrepancy between actual values and

predicted values, given by [13]:

$$MRE = \frac{|\ actual\ value - predicted\ value\ |}{actual\ value} \tag{17}$$

In this study, the maximum value of MRE (Max.MRE) is used. The Max.MRE measures the maximum relative discrepancy, which is equivalent to the maximum error relative to the actual effort in the prediction. The mean of MRE, the mean magnitude of relative error (MMRE):

$$MMRE = \frac{1}{n}\sum_{i=1}^{i=n} MRE_i \tag{18}$$

is also used. MMRE measures the average relative discrepancy, which is equivalent to the average error relative to the actual effort in the prediction. Sometimes MMRE is expressed in %. However, this study follows the difinition given in Equation 18 and does not express MMRE in %.

Pred is a measure of what proportion of the predicted values have MRE less than or equal to a specified value, given by [9]:

$$Pred(q) = \frac{k}{n} \tag{19}$$

where q is the specified value, k is the number of cases whose MRE is less than or equal to q, and n is the total number of cases in the dataset. In this paper, pred(0.25) and pred(0.30) are used since those two pred measures are commonly used in the software effort prediction literature.

In order for an effort prediction model to be considered accurate, $MMRE \leq 0.25$ [6] and/or either $pred(0.25) \geq 0.75$ [6] or $pred(0.30) \geq 0.70$ [14] is suggested in the literature. On the other hand, there is a growing concern about MRE, since MRE is biased [19] and not always reliable as a predictive accuracy measure [10]. However, MRE has been the de facto standard in the software effort prediction literature and no alternative standard exists at present. Hence, MRE is still used in this study. However, in addition to MRE, the absolute residual measures are used. This is because the absolute residual measures, in particular the SD Ab.Res., are shown to be a better measure than MRE for model comparison [10].

6 Validation method

6.1 The whole ISBSG dataset

As was mentioned in Section 4, the BR and BMVN models were first calibrated using Calibration-W subset, which consists of 115 systems randomly sampled from the ISBSG dataset. The remaining 57 systems in the ISBSG dataset form the first validation dataset. We refer to this validation dataset as Validation-W in the remainder of this paper. Using Validation-W, the models are evaluated and compared in terms of the predictive accuracy measures defined in Section 5. Then, the models' predictive accuracy is compared with the corresponding linear regression model:

$$LogEFFORT = \beta_0 + \beta_1 * LogFP \tag{20}$$

The corresponding linear regression model is built from Validation-W using a software package SPSS.

Secondly, the BR and BMVN models were calibrated using Calibration-W-S1 $\sim$ -S4, which consist of five systems each. This resulted in the corresponding four different validation datasets, each of which consists of the remaining 167 systems. We refer to those validation datasets as Validation-W-S1 $\sim$ -S4. Using Validation-W-S1 $\sim$ -S4, the models' overall predictive accuracy are evaluated and compared in the way similar to that of Validation-W. The overall predictive accuracy is the average values of Validation-W-S1 $\sim$ -S4, that is, the average of $167 \times 4 = 668$ random samples, which were independently drawn from the 172 systems in the ISBSG dataset.

6.2 Clusters of systems

As was mentioned in Section 4, the BR and BMVN models were first calibrated independently using Calibration-C-1 $\sim$ -7. This resulted in the corresponding seven validation datasets. We refer to those validation datasets as Validation-C-1 $\sim$ -7 respectively. The leave-one-out method produces the validation dataset that is exactly the same as the cluster itself. Hence Validation-C-1 $\sim$ -7 consist of 12, 27, 17, 24, 12, 37 and 10 systems respectively. Using Validation-C-1 $\sim$ -7, the models are evaluated and compared independently within each cluster, in the way similar to Validation-W.

Secondly, the BR and BMVN models were calibrated using Calibration-C-1-S1 $\sim$ -S18 in Cluster 1, Calibration-C-4-S1 $\sim$ -S8 in Cluster 4, and Calibration-C-

Model	MRE		Pred		Ab.Res.		
	Max.	Mean	0.25	0.30	Sum	Med.	SD
BR	8.33	1.17	0.23	0.26	119590	1078	2976
BMVN	8.30	1.17	0.23	0.26	119566	1071	2988
R	8.31	1.17	0.23	0.26	119596	1078	2976

Table 3
Predictive accuracy of a large calibration subset

7-S1 $\sim$ -S23 in Cluster 7. This resulted in the corresponding Validation-C-1-S1 $\sim$ -S18, Validation-C-4–S1 $\sim$ -S8 and Validation-C-7-S1 $\sim$ -S23. Validation-C-1-S1 $\sim$ -S18 consist of 9 systems each, since Cluster 1 consists of 12 systems and three among the 12 systems were in Calibration-C-1-S1 $\sim$ -S18. Similarly, Validation-C-4-S1 $\sim$ -S8 consist of 21 systems each and Validation-C-7-S1 $\sim$ -S23, 7 systems. The models are evaluated and compared in the way similar to that of Validation-W-S1 $\sim$ -S4. That is, the average values of all the validation datasets in the same cluster are taken as the overall predictive accuracy. Hence, the overall predictive accuracy in Cluster 1 is the average of $9 \times 18 = 162$ independent random samples, which were drawn from the 12 systems in the cluster. Similarly, the overall predictive accuracy of Cluster 4 is the average of $21 \times 8 = 168$ random samples, and Cluster 7, the average of $7 \times 23 = 161$ random samples.

7 Model evaluation

7.1 The whole ISBSG dataset

Table 3 shows the predictive accuracy measure values achieved in Validation-W by the BR and BMVN models, together with the corresponding linear regression model. We refer to the linear regression model simply as R for convenience in this section. It should be noted that the Ab.Res. values are measured in hours. That is, the predicted LogEFFORT was transformed back to the EFFORT value measured in hours, in order to calculate the accuracy values presented. Table 3 shows that the three models' predictive accuracy in Validation-W are almost identical. That is, the predictive accuracy of the two Bayesian statistical models are equivalent to the corresponding linear regression model, when the calibration subset consists of 115 systems. In order to confirm this finding, Wilcoxon signed-rank tests were performed on the MRE and Ab.Res. values. The Wilcoxon signed-rank test tests for a difference between two related samples, assuming no condition on the population distri-

Model	MRE		Pred		Ab.Res.		
	Max.	Mean	0.25	0.30	Sum	Med.	SD
BR	10.69	0.91	0.16	0.20	474510	968	6324
BMVN	9.71	0.85	0.19	0.23	396904	879	4608
R	10.79	0.92	0.16	0.20	478125	970	6457

Table 4
Overall predictive accuracy of small calibration subsets

bution. Other tests that assume normality on the population distribution are not appropriate, since MRE and Ab.Res. distributions are, in general, known to be not normal. The Wilcoxon signed-rank tests for pair-wise comparisons between the BR and R models, and the BMVN and R models, confirmed no evidence of a difference between each pair. Hence the tests support the above finding.

Table 4 shows the models' overall predictive accuracy, which are the average values of Validation-W-S1 $\sim$ -S4. Table 4 shows that the overall predictive accuracy of the BR model is slightly, and the BMVN model is considerably better than the linear regression model. These findings were confirmed by the Wilcoxon signed-rank tests, which showed strong evidence of a difference between the BR and R models, and the BMVN and R models. Hence it is concluded that the overall predictive accuracy of the two Bayesian statistical models are better than the corresponding linear regression model, when the calibration subset consists of only 5 systems. Table 4 also shows that the overall predictive accuracy of the BMVN model is better than the BR model. This finding was also confirmed by another Wilcoxon signed-rank test, which showed strong evidence of a difference between those two models. Hence it is concluded that the overall predictive accuracy of the BMVN model is better than the BR model, when the calibration dataset consists of only 5 systems.

7.2 Clusters of systems

Table 5 shows the predictive accuracy measure values achieved in Validation-C-1 $\sim$ -7. The first column in this table indicates the number of cluster. Table 5 shows that the three models' predictive accuracy in each cluster are very similar. The Wilcoxon signed-rank tests confirmed no evidence of a difference between the BR and R models, and the BMVN and R models, for all the clusters except Cluster 2. In addition, in Cluster 2 where the differences are found significant, the MMRE and three Ab.Res. values suggest that the BR and BMVN models are better than the linear regression model. Hence, it is

	Model	MRE		Pred		Ab.Res.		
		Max.	Mean	0.25	0.30	Sum	Med.	SD
1	*BR*	2.12	0.55	0.25	0.25	6150	335	525
	BMVN	2.07	0.55	0.25	0.33	5796	317	489
	R	2.14	0.55	0.25	0.25	6118	337	518
2	*BR*	2.24	0.58	0.44	0.44	30913	708	1688
	BMVN	2.26	0.58	0.44	0.44	30905	715	1685
	R	2.23	0.60	0.44	0.48	31931	736	1690
3	*BR*	7.70	0.97	0.29	0.35	26951	824	2139
	BMVN	7.70	0.97	0.35	0.41	26310	860	2168
	R	7.71	0.98	0.29	0.35	26963	827	2129
4	*BR*	4.96	0.81	0.29	0.29	42272	869	2139
	BMVN	4.71	0.83	0.29	0.29	43046	988	2193
	R	4.98	0.81	0.29	0.29	42147	871	2126
5	*BR*	4.23	1.55	0.08	0.08	49651	2177	4854
	BMVN	5.24	1.55	0.08	0.08	41574	2027	3797
	R	4.26	1.56	0.08	0.08	49826	2173	4893
6	*BR*	11.14	1.38	0.16	0.22	74463	1135	2814
	BMVN	11.29	1.38	0.16	0.22	74094	1060	2832
	R	11.15	1.38	0.16	0.22	74441	1129	2816
7	*BR*	1.25	0.41	0.22	0.33	7266	776	514
	BMVN	1.39	0.43	0.33	0.33	7221	910	362
	R	1.24	0.41	0.22	0.33	7251	766	530

Table 5
Predictive accuracy of large calibration subsets

concluded that the predictive accuracy of the two Bayesian statistical models are at least as good as the corresponding linear regression model within each cluster.

Table 6 shows the models' overall predictive accuracy in Validation-C-1-S1 $\sim$ -S18, -C-4-S1 $\sim$ -S8 and -C-7-S1 $\sim$ -S23. The first column in this table indicates the number of cluster. Table 6 shows that the overall predictive accuracy of the two Bayesian statistical models are considerably better than the linear regression model in all the three clusters. In addition, the overall predictive

	Model	MRE		Pred		Ab.Res.		
		Max.	Mean	0.25	0.30	Sum	Med.	SD
1	*BR*	3032.58	338.59	0.22	0.25	8904934	990	2966754
	BMVN	1.93	0.66	0.30	0.38	4427	334	490
	R	5650.82	629.81	0.22	0.25	16584698	1129	5522245
4	*BR*	64.15	8.97	0.09	0.11	1538907	3076	186894
	BMVN	3.82	1.00	0.13	0.16	55414	1605	2758
	R	69.55	9.78	0.10	0.11	1705868	3133	209065
7	*BR*	$6.99E+9$	$1.21E+9$	0.23	0.27	$3.37E+13$	286957748	$9.95E+12$
	BMVN	14.08	2.42	0.30	0.35	7533	1077	669
	R	$1.31E+11$	$2.24E+10$	0.23	0.26	$6.24E+14$	$1.56E+9$	$1.87E+14$

Table 6
Overall predictive accuracy of small calibration subsets

	Comparison	MRE	Ab.Res.
1	*BR-R*	0.000**	0.002**
	BMVN-R	0.248	0.045*
	BR-BMVN	0.300	0.054
4	*BR-R*	0.000**	0.000**
	BMVN-R	0.010*	0.020*
	BR-BMVN	0.012*	0.023*
7	*BR-R*	0.007**	0.086
	BMVN-R	0.259	0.005**
	BR-BMVN	0.294	0.007**

Table 7
P-values for pair-wise comparisons

accuracy of the BMVN model is considerably better than the BR model. The following Table 7 shows the p-values from the Wilcoxon signed-rank tests for pair-wise comparisons. The first column in the table indicates the number of cluster. The * symbol after a p-value indicates that the value is significant at the 5% level and the ** symbol, at the 1% level.

In Cluster 1, Table 7 confirms a significant difference between the BR and R

models, and a significant difference in the Ab.Res. values between the BMVN and R models. These results support the above finding that the overall predictive accuracy of the two Bayesian statistical models are better than the linear regression model. Between the BR and BMVN models, the p-value of 0.054 provides weak evidence of a difference in the Ab.Res. values. Although this result only is not conclusive, it also supports another finding that the BMVN model seems to outperform the BR model.

On the other hand, in Cluster 4, Table 7 confirms a significant difference in each pair. Hence, these results support the findings that the overall predictive accuracy of the two Bayesian statistical models are better than the linear regression model and that the BMVN model outperforms the BR model.

In Cluster 7, Table 7 confirms a significant difference in the MRE values between the BR and R models, and a significant difference in the Ab.Res. values between the BMVN and R models. These results again support the finding that the overall predictive accuracy of the two Bayesian statistical models are better than the linear regression model. Between the BR and BMVN models, the strongly significant p-value of 0.007 provides strong evidence of a difference in the Ab.Res. values. Hence this supports another finding that the BMVN model outperforms the BR model.

7.3 Discussion

The above findings that the predictive accuracy of the two Bayesian statistical models are in general equivalent to the linear regression model, however, become significantly better when the calibration subset becomes very small, support the expectation of this study, which is, Bayesian statistical models based on good prior knowledge can achieve good predictive accuracy even if only a very small amount of empirical data is used for calibration. However, more case studies would be needed in order to extend these findings to software datasets, in general.

In this study it is also observed that the predictive accuracy achieved by any of the models are low, in comparison with the values in Section 5, which are suggested for the models to be considered accurate. However, the author thinks these seemingly low predictive accuracy are not due to the models' inadequacy. This is because the influences of all the candidate variables were tested in the preliminary data analysis in Section 4, and the possible influences of the non-significant variables were removed by using the clusters in the second half of this study. In addition, the significant regression effects of the linear regression models indicate a linear relationship between LogFP and LogEFFORT, and hence the linear model should be appropriate. One possible explanation of

the low accuracy is instead that the single predictor variable FP was not able to explain the variance of EFFORT fully in the ISBSG dataset, although its influence is highly significant. This suggests that there may exist one or more other unknown influential variables on EFFORT in the ISBSG dataset.

Another finding in this study that the BMVN model outperforms the BR model, requires further investigation. The author currently suspects that this happens only in the dataset, where effort can be explained from the combined influence of all predictor variables rather than from the added influences of the individual variables. This is because the BMVN model makes use of the combined influence, while the BR model makes use of the added influences. However, this is just one possibility and there would be some other possibilities that need to be explored. Hence, investigating the reason of the BMVN model's better performance is certainly an interesting direction of future studies.

8 Conclusions

This paper presents two new Bayesian statistical models, in order to predict development effort of software systems in the ISBSG dataset. They are a Bayesian linear regression (BR) model and a Bayesian multivariate normal distribution (BMVN) model. Both models are calibrated using various subsets randomly sampled from the dataset. Then, the models' predictive accuracy is evaluated using the corresponding validation subsets, which consist of only the cases unknown to the models. The predictive accuracy is measured in terms of the absolute residuals and magnitude of relative error. They are also compared with the corresponding linear regression models.

The findings are that the predictive accuracy of the two Bayesian statistical models are in general equivalent to the linear regression model, however, become significantly better when the calibration subset becomes very small. Another finding is that the BMVN model outperforms the BR model. Although more studies would be needed to extend these findings to other software datasets, they should justify the use of Bayesian statistical models in software development effort prediction, in particular when the system of interest has only a very small amount of historical data. The author also expects the results from the related studies in future enable Bayesian statistical models to take a leading role in software effort prediction.

In addition to conducting more case studies using the above and/or new Bayesian statistical models in different software datasets, another interesting direction of future studies would be to investigate the reason of the BMVN model's better performance. The author thinks it could be achieved by a designed experiment using simulated datasets, where the predictive relation-

ships of variables are controlled. The use of simulated datasets would be well justified since it is very less likely to realize this designed experiment using real software datasets. The results from these studies are expected to provide some important insight into software effort prediction for both practitioners and researchers.

References

[1] C.G. Bai. Bayesian network based software reliability prediction with an operational profile. *The Journal of Systems and Software*, 77:103–112, 2005.

[2] C.G. Bai, Q.P. Hu, M. Xie, and S.H. Ng. Software failure prediction based on a Markov bayesian network model. *The Journal of Systems and Software*, 74:275–282, 2005.

[3] J. Baik, B. Boehm, and B.M. Steece. Disaggregating and calibrating the CASE tool variable in COCOMO II. *IEEE Transactions on Software Engineering*, 28(11):1009–1022, 2002.

[4] S. Chulani, B. Boehm, and B.M. Steece. Bayesian analysis of empirical software engineering cost models. *IEEE Transactions on Software Engineering*, 25(4):513–583, 1999.

[5] P. Congdon. *Bayesian Statistical Modelling*. John Wiley & Sons., 2001.

[6] S.D. Conte, H.E. Dunsmore, and V.Y. Shen. *Software Engineering Metrics and Models*. Benjamin/Cummings Publishing Company, 1986.

[7] C. Fan and Y. Yu. BBN-based software project risk management. *The Journal of Systems and Software*, 73:193–203, 2004.

[8] N. Fenton and M. Neil. A critique of software defect prediction models. *IEEE Transactions on Software Engineering*, 25(5):675–689, 1999.

[9] N.E. Fenton and S.L. Pfleeger. *Software Metrics:A Rigorous & Practical Approach*. PWS Publishing Company, second edition, 1997.

[10] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit. A simulation study of the model evaluation criterion mmre. *IEEE Transactions on Software Engineering*, 29(11):985–995, 2003.

[11] P.J. Green. A primer on markov chain monte carlo. In O.E. Barndorff-Nielsen, D.R. Cox, and C. Klüppelberg, editors, *Complex Stochastic Systems*, chapter 1, pages 1–62. Chapman & Hall/CRC, 2001.

[12] F.V. Jensen. *Bayesian Networks and Decision Graphs*. Springer–Verlag New York, 2001.

[13] B.A. Kitchenham, L.M. Pickard, S.G. MacDonell, and M.J. Shepperd. What accuracy statistics really measure. *IEE Proceedings–Software*, 148(3):81–85, 2001.

[14] S.G. MacDonell. Establishing relationships between specification size and software process effort in case environment. *Information and Software Technology*, 39:35–45, 1997.

[15] J. Moses. Bayesian probability distributions for assessing measurement of subjective software attributes. *Information and Software Technology*, 42:533–546, 2000.

[16] M. Neil, N. Fenton, and L. Nielsen. Building large-scale bayesian networks. *The Knowledge Engineering Review*, 15(3):257–284, 2000.

[17] P.C. Pendharkar, G.H. Subramanian, and J.A. Rodger. A probabilistic model for predicting software development effort. *IEEE Transactions on Software Engineering*, 31(7):615–624, 2005.

[18] I. Stamelos, L. Angelis, P. Dimou, and E. Sakellaris. On the use of Bayesian belief networks for the prediction of software productivity. *Information and Software Technology*, 45:51–60, 2003.

[19] E. Stensrud, T. Foss, B.A. Kitchenham, and I. Myrtveit. An empirical validation of the relationship between the magnitude of relative error and project size. In *Proceedings of the 8th IEEE Symposium on Software Metrics (METRICS'02)*, pages 3–12, 2002.

[20] B. Stewart. Predicting project delivery rates using the Naive–Bayes classifier. *Journal of Software Maintenance and Evolution: Research and Practice*, 14:161–179, 2002.

[21] C. van Koten and A.R. Gray. An application of bayesian network for predicting object-oriented software maintainability. *Information and Software Technology*, in press, 2005.

[22] D.A. Wooff, M. Goldstein, and F.P.A. Coolen. Bayesian graphical models for software testing. *IEEE Transactions on Software Engineering*, 28(5):510–525, 2002.