Key words:
defect prediction, defect types, Bayesian Nets

Łukasz RADLIŃSKI*‡
Norman FENTON‡
David MARQUEZ‡
Peter HEARTY‡

EMPIRICAL ANALYSIS OF SOFTWARE DEFECT TYPES

There have been numerous models predicting software defect counts or probability that a given software part (e.g. module) is defective. The need for such models is unquestionable. However, in many cases software managers would be interested not just in number of defects which are likely to occur in software but also the number of defects by severity (such as extreme, major, minor). In this paper we present results of analysing data from the ISBSG database of software projects focusing on identification of factors influencing types of defects as well as the nature of these influences. We also demonstrate a Bayesian Net (BN) – a causal model that enables us to predict defect types using a combination of quantitative and qualitative factors. This model incorporates results of analysis based on several sources of publicly available data about the past projects, mainly ISBSG dataset and other literature. This model can be combined with a previously developed BN model that integrates resource and defect prediction.

1. INTRODUCTION

Traditional defect prediction models do not distinguish types of defects, treating them all equally. A defect means that client or user does not receive what they expect.

Our ultimate aim is to build a model that will predict not just the total number of defects (although this itself is a demanding task) but also which will categorize defects by their severity. For this we mainly use the ISBSG dataset of software projects because this is the only publicly available dataset we are aware of containing relatively high

* University of Szczecin, ul. Mickiewicza 64, 71-101 Szczecin, Poland
‡ Queen Mary College, University of London, Mile End Road, London E1 4NS, United Kingdom

number of observations of categorized defect data. The aim of this paper is to identify factors that influence the occurrence of different numbers of defects of a particular type.

ISBSG defines defect as 'A failure of some part of an application' [8]. They distinguish three types of defects in the dataset depending on their severity:

1. Minor – 'A minor defect does not make the application unusable in any way, (e.g. a modification is required to a screen field or report)'.
2. Major – 'A major defect causes part of the application to become unusable'.
3. Extreme – 'A failure of some part of an application that causes the application to become totally unusable' [8].

In Section 2 we present the dataset that we used for the analysis. In Section 3 we analyze factors influencing the occurrence of different types of defects.

The Bayesian Net (BN) model, which we propose in Section 4, predicts proportions of different types of defects. It includes results from the analysis of the ISBSG dataset and software engineers' knowledge. However, it does not predict number of defects of specific types directly. To achieve this our model can be combined with any other model predicting total number of defects. In particular, this could be other BN models such as [9, 11], or models like in [2, 4, 5].

2. ISBSG DATASET DESCRIPTION

The ISBSG R9 database [6] contains data about 3024 software projects described by 99 variables. We filtered data by leaving only those projects for which 'data quality' was classified as 'A' or 'B' (meaning respectively 'very good' and 'good' quality) as suggested in [7]. As a result we got a dataset with 2790 projects. For our analysis we could not use all of them because only 467 projects contained relevant information about defects.

The main three variables we analyzed were the proportions of specific defect types (minor, major, extreme) in total number of defects in a project. Because for some projects no defects were found, proportions of defects of different types could not be calculated for them. As a result for the analysis we used a dataset of 311 software projects containing at least one defect, for which a proportion of number of particular defect types in total number of defects could be calculated.

Table 1 illustrates the mean and median values for proportions of different defect types together with distributions that fits the data best. We use these distributions in developing a causal model (Section 4).

For comparison we also included data from other publicly available defect databases, namely Eclipse and Mozilla [3, 10]. The different proportions of defects in these compared to ISBSG is due to a different type of defect classification. Eclipse and Mozilla use 7, rather than 3, categories. To create the table we merged some of these. Since the

Eclipse and Mozilla databases do not contain as many factors and observations as the ISBSG database and we did not use them in the subsequent analysis.

Table 1. Descriptive statistics and best-fit distributions for proportions of different defect types

Proportions of defects:	ISBSG		Eclipse		Mozilla	
	Mean / Median	Best-fit distribution	Mean / Median	Best-fit distribution	Mean / Median	Best-fit distribution
Minor	0.49 0.53	Beta (2.35, 1.38)	0.21 0.18	Extreme (0.16, 0.09	0.25 0.23	Gamma (3.77, 0.07)
Major	0.46 0.30	Beta (1.97, 4.70)	0.72 0.74	Beta (17.18, 6.52)	0.70 0.72	Beta (7.02, 3.12)
Extreme	0.05 0.00	Log Normal (-1.78, 1.09)	0.07 0.05	Log Normal (-2.94, 0.81)	0.05 0.03	Log Normal (-3.16, 0.72)

Fig. 1 illustrates the histograms of the proportions of different types of defects in total number of defects. They show that among all the projects there were projects:

- in which all defects were of a single type or
- containing various types of defects.

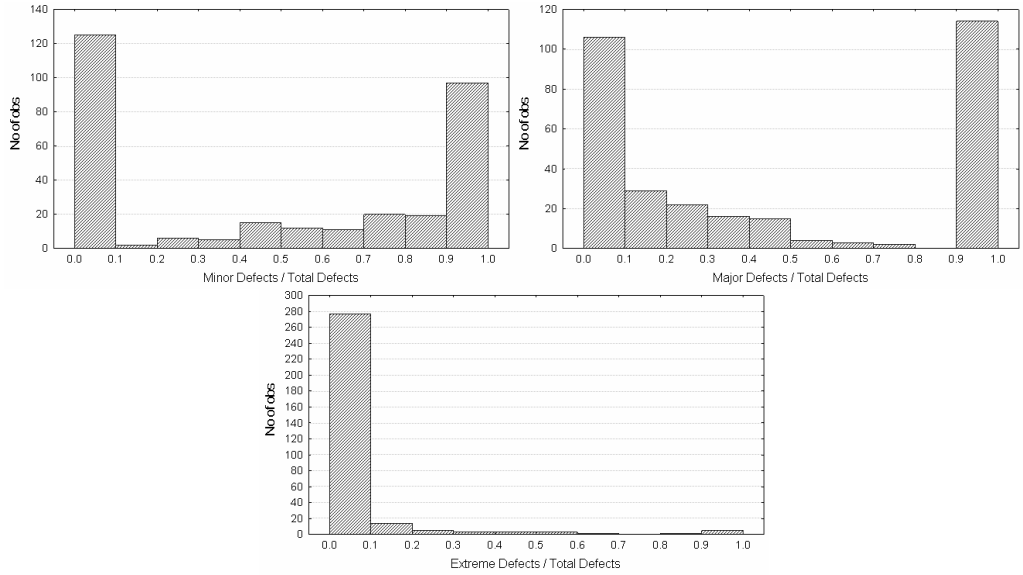

Fig. 1. Histograms of proportions of different types of defects in total number of defects

We can observe two dominant values on the figure: 0 and 1. The value of '0' means that in a project there were defects but of different type than the histogram presents. The

value of ‘1’ means that all of defects in a project were of this particular type. We can observe on the histograms that when all of the defects from a project were classified to a single category they were usually either minor or major. In only 5 projects were all of the defects classified as extreme.

We found two main reasons for the fact that some projects contained only one type of defect:

- these projects are usually smaller and the total number of defects is smaller compared with projects containing various types of defects.
- data provided by software companies might not be reliable in a sense that only one type of defect might have been reported. For example there was a project in which 285 defects were reported and all of them were major.

3. FACTORS INFLUENCING TYPES OF DEFECTS

The dataset used in the analysis contained both numeric and descriptive data. We prepared a list of potential factors which, in our belief, could influence predictor variables. We started with identification of numeric factors influencing proportions of different defect types.

The Pearson product-moment correlation coefficient is most widely used for calculating correlations, but it assumes normality in the distribution of analyzed variables. As our predictive variables were clearly not normally distributed we use Spearman’s rank correlation coefficient instead to calculate correlations between our predictor and predictive variables. We present the results of this analysis in Table 2. We observe that:

- There is a high negative correlation (-0.93) between proportion of minor defects and proportion of major defects. This means that as proportion of minor defects increases (decreases), the proportion of major defects decreases (increases) by around the same value.
- We did not find any correlation between proportion of extreme defects and proportions of other types of defects.
- Proportion of effort on specification and number of business units serviced by the software application are the highest correlated factors with proportions of minor and major defects.
- Proportion of effort on implementation is the highest correlated (0.44) factor with proportion of extreme defects.

Table 2. Numeric predictor variables influencing proportions of different defect types

Predictive variables / Predictor variables	Proportion of minor defects	Proportion of major defects	Proportion of extreme defects
Adjusted Function Points	0.06	-0.03	**0.17**
Minor Defects / Total Defects	1	**-0.93**	-0.08
Major Defects / Total Defects	**-0.93**	1	-0.11
Extreme Defects / Total Defects	-0.08	-0.11	1
Normalised Productivity Rate (Normalised Work Effort / Adjusted Function Points)	**-0.12**	**0.14**	**-0.15**
Project Inactive Time	**0.20**	**-0.24**	0.11
Proportion of Effort on Plan	0.12	-0.09	0.11
Proportion of Effort on Specification	**0.40**	**-0.38**	0.15
Proportion of Effort on Design	0.62	-0.62	.
Proportion of Effort on Build	**-0.16**	0.14	0.07
Proportion of Effort on Test	0.08	-0.09	-0.02
Proportion of Effort on Implementation	-0.19	0.12	**0.44**
Total Defects / Function Point	**-0.15**	**0.16**	**0.25**
User Base – Business Units	**0.43**	**-0.44**	0.12
User Base – Locations	**0.20**	**-0.20**	-0.11

numbers in 'bold' indicate statistically significant at $p < 0.05$

The mainly low correlation coefficient values mean that we cannot identify relationship between the predictor and predictive variables. So we decided to use also nominal variables in the analysis as potential factors influencing proportion of different defect types. We prepared a list of factors, which potentially could impact proportions of defect types. For this analysis we used Kruskal-Wallis one-way ANOVA (analysis of variance) which does not assume normality of predictive variables. We visually analyzed box-plots and histograms of specific defect types categorized by the states of descriptive factors. If box-plots and histograms varied among different states of descriptive factor we assumed that such factor was correlated with proportion of particular defect type. To confirm our suspicions we also checked if the Kruskal-Wallis test was statistically significant ($p < 0.05$).

Some states of descriptive factors used in this analysis had no matching observations of proportions of defect types. These states were automatically excluded from our analysis by the statistical tool used. Next, in around half of the descriptive factors we found some states with a small number of observations of defect types. For these variables we kept only the states for which we had at least 8 observations.

We present results from this analysis in Table 3. We observe that:

- Five descriptive factors were correlated with all defect types.

- Five descriptive factors were correlated with proportion of one or two defect types.
- Three descriptive factors were not correlated with proportion of any defect type.

Table 3. Nominal predictor variables influencing proportions of different defect types

Predictive variables / Predictor variables	Proportion of minor defects	Proportion of major defects	Proportion of extreme defects
Application Type	+	+	+
Architecture	–	–	–
Business Area Type	+	+	+
CASE Tool Used	+	+	+
Client-Server	–	+	–
Development Platform	–	–	–
Development Type	–	–	–
Intended Market	+	+	+
Language Type	+	+	–
Organisation Type	+	+	+
Package Customisation	+	–	+
Type of Server	–	–	+
Used Methodology	–	–	+

'+' indicates that statistically significant (at $p < 0.05$) correlation was found

'–' indicates that statistically significant (at $p < 0.05$) correlation was not found

4. BAYESIAN NET FOR PREDICTING DEFECT TYPES

In our BN (Fig. 2) for predicting proportions of different types of defects we used factors that we identified as important during the analysis discussed in the previous Section. We also added other subjectively selected factors which we believe are important from our software engineering experience. They are expressed on a 5-point ranked scale from 'very low' to 'very high'. They express the quality of different software development activities: specification, coding, testing, implementation. Other factors are: 'project scale & complexity', 'customer communications', 'staff quality'.

The model uses prior probability distributions from Table 1. Then adjustments are estimated according to the observations provided by the user for predictor variables. Prediction for proportion of specific defect type is calculated by adjusting prior probability distribution using 'adjustment' variables.

Because of the model structure, it might be possible that for certain scenarios the sum of proportions of all defect types is not exactly 1. To avoid this, our model contains a

hidden constraint node. Its probability table is an expression summing proportions of different defect types. In each scenario it must have an observation with the value '1'. As a result proportions of defects are adjusted during calculations to always sum up to 1.

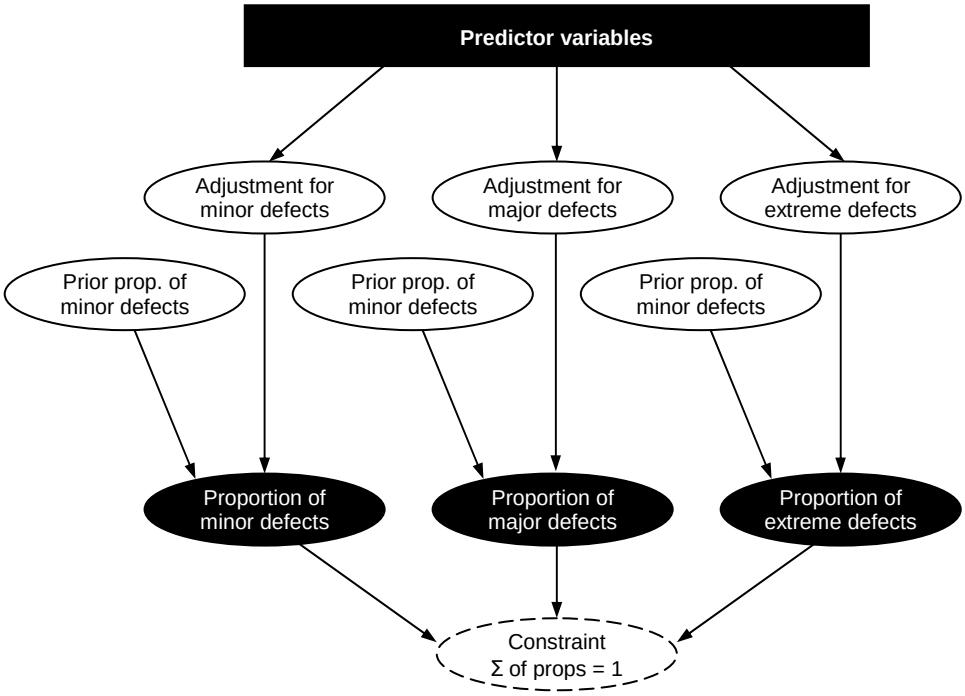

Fig. 2. Bayesian Net for defect types prediction

Predictions for number of different types of defects (Fig. 3) are estimated by combining:

- the output of our model: proportions of different types of defects
- the value for total number of defects.

Probability distributions for number of different types of defects are Binominal with:

- number of trials = total number of defects,
- probability of success = proportion of specific defect type.

Total number of defects is an input in this model. There are other BN models [9, 11] that output number of defects, so the idea is to use those models to generate the input here. Hence, total number of defects will be in the form of a probability distribution. We can also use non-BN models [2, 4, 5] to predict total number of defects, in which case they will be point values (not distributions) and we enter these as observations in our model.

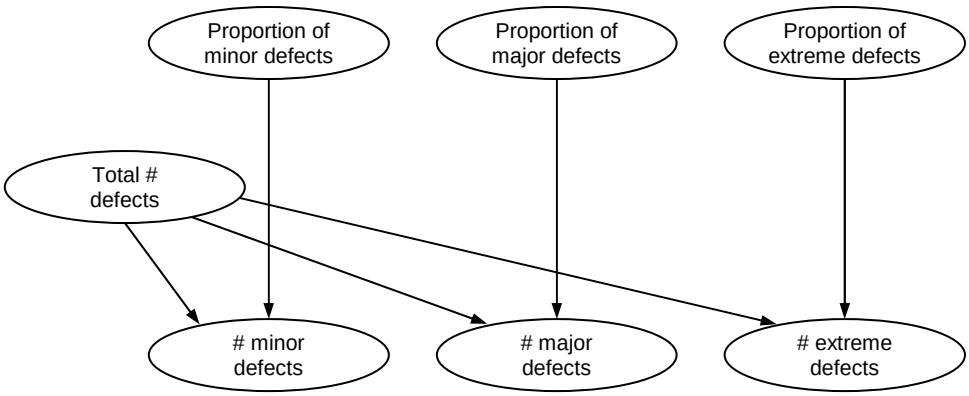

Fig. 3. Predicting number of different types of defects in integrated model

BNs can be calculated without the need for providing observations for all predictive variables. This is a useful feature of such models because often we do not know values for all predictive variables. If we do not provide an observation for a certain variable, the prior probability distribution is used.

5. SUMMARY

We analysed an extensive empirical dataset of software projects to identify factors influencing proportions of different defects types categorized by severity: minor, major, severe. We found very small correlations between numeric factors and our predictive variables using Spearman's rank correlation coefficient. To complement this we also performed Kruskal-Wallis one-way ANOVA in which we identified some descriptive factors influencing proportions of different defect types.

We also developed a BN model for predicting different types of defects. In this model we included variables identified as important factors for different defects types. Using expert knowledge we added other variables not present in the analyzed dataset, but which are also important predictors for proportions of different types of defects. We will continue this work by:

- extending the analysis by analyzing the impact of different combinations of factors (not single factors like in the current analysis) on proportions of defect types,
- analyzing different datasets of software projects, mostly for open-source projects such as Apache, Eclipse, Mozilla [1, 3, 10].
- extending the model with the results of newer analyses.
- validating the model against empirical data and axioms and theories in software engineering.

REFERENCES

[1] APACHE SOFTWARE FOUNDATION, *ASF Bugzilla*, 2007, http://issues.apache.org/bugzilla/report.cgi.

[2] CHULANI S., BOEHM B., *Modelling Software Defect Introduction and Removal: COQUALMO (COnstructive QUAlity MOdel)*, Technical Report USC-CSE-99-510, University of Southern California, Center for Software Engineering, 1999.

[3] ECLIPSE, *Eclipse Bugs*, 2007, https://bugs.eclipse.org/bugs/.

[4] GAFFNEY J.R., *Estimating the Number of Faults in Code*, IEEE Trans. Software Engineering, Vol. 10, No. 4, 1984. 141–152.

[5] GOKHALE S.S., LYU M.R., *Regression Tree Modelling for the Prediction of Software Quality*, Proc. ISSAT International Conference on Reliability and Quality in Design, Anaheim, 1997. 31–36.

[6] ISBSG, *Estimating, Benchmarking & Research Suite Release 9*, International Software Benchmarking Standards Group, 2005.

[7] ISBSG, *ISBSG Comparative Estimating Tool V4.0 – User Guide*, International Software Benchmarking Standards Group, 2005.

[8] ISBSG, *Glossary of Terms*, V5.9.1, International Software Benchmarking Standards Group 28/02/2006.

[9] MODIST, *Models of Uncertainty and Risk for Distributed Software Development*, EC Information Society Technologies Project IST-2000-28749, www.modist.org, 2003.

[10] MOZILLA.ORG, *Bugzilla@Mozilla*, 2007, https://bugzilla.mozilla.org/.

[11] RADLIŃSKI Ł., FENTON N., NEIL M., MARQUEZ D., *Improved Decision-Making for Software Managers Using Bayesian Networks*, manuscript submitted to: IFIP Central and East European Conference on Software Engineering Techniques, Poznań 2007.