

# Predicting Software Defect Function Point Ratios Using a Bayesian Belief Network

James B. Dabney  
Systems Engineering Program  
University of Houston – Clear Lake  
Houston, Texas  
dabney@cl.uh.edu

Gary Barber and Don Ohi  
L3 Communications Titan Group.  
NASA IV&V Facility  
Fairmont, West Virginia  
{gary.barber, don.ohi}@L-3Com.com

## Abstract

An important component of software process predictive modeling is computation of a suitable measure of defect density based on project characteristics that can be estimated early in the project lifecycle. This paper presents a Bayesian belief network-based approach to estimating defect density in the form of defect function point ratios, calibrated empirically.

## Keywords

Verification and validation, return on investment, defect leakage, cost modeling.

## 1 Introduction

A common software process model component is prediction of defect density statistics based on project and process characteristics which can be ascertained early in the project lifecycle. This paper presents a methodology to compute defect discovery profiles for the software developer and an independent verification and validation (IV&V) agent suitable for use in a software IV&V return on investment (ROI) model [1].

Section 2 describes briefly the IV&V direct ROI model. Section 3 describes the IV&V direct ROI predictive model and section 4 describes the model calibration. Section 5 presents conclusions and suggests future work.

## 2 IV&V Direct ROI Model

A standard management measure for determining the worth of an investment is return on investment (ROI), also known as benefit/cost ratio [2]. ROI is widely recognized as an important measure and is a required metric of certain government-supported projects [2]. In order to compute the ROI of software independent verification and validation (IV&V) a method to compute IV&V ROI based on project records was developed.

There are many components of the total cost savings due to IV&V. For example, benefits include reduced development cost, increased confidence in the final product, improved quality, reduced risk, and improved safety. Most of the benefits are difficult to quantify. However, one benefit, reduced development cost due to early defect detection, can be estimated relatively accurately. We refer to this component of ROI as *direct ROI*. The ROI resulting from other IV&V benefits is termed *indirect ROI* and is more difficult to compute. In previous work, we have described techniques to compute

direct [1] and indirect [3] ROI. This paper is concerned only with direct ROI.

We define direct ROI [1] as the ratio  $(C_x - C_i)/C_{IVV}$ , where  $C_x$  is the project development cost without IV&V,  $C_i$  is the project development cost with IV&V, and therefore the difference  $\delta C_r = (C_x - C_i)$  is reduction in development cost due to early issue identification by the IV&V team.  $C_{IVV}$  is the cost of the IV&V effort. Cost can be expressed in any consistent unit. Typically, equivalent person-month (EPM) is the most convenient unit.

The basis of our approach to computing  $\delta C_r$  is to compute rework cost for the actual with-IV&V data from project records and to conservatively estimate the rework cost without IV&V by assuming that issues identified by IV&V would have been discovered later in the project by the developer with the same probability distribution as other issues discovered by the developer. Rework costs can in some cases be obtained directly from project records [1]. More commonly rework costs must be computed using a software cost estimation tool such as COCOMO-II [4], [5].

The direct ROI model is depicted graphically in Figure 1. The inputs to the model are characteristics of issues discovered by IV&V and the developer (issue type, phase found, cost-to-fix) and developer and IV&V project costs.

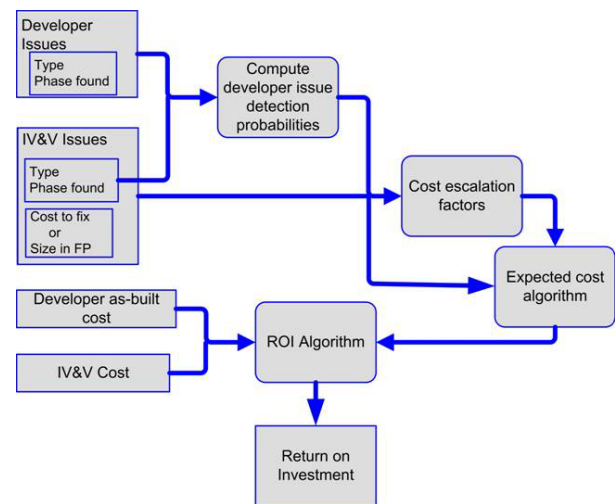


Figure 1: Computing direct ROI

### 3 Predicting Direct ROI

The direct IV&V ROI model [1] provides the means to compute ROI for completed IV&V projects. The model requires as inputs developer and IV&V costs (typically in equivalent person months (EPM)), software product size (measured in source lines of code (SLOC) or function points (FP)[6]), and measures of defect detection by the developer and IV&V for each type of issue (requirements, design, code, test, integration) and each development phase (typically in cost-to-fix EPM or issue size in FP).

The complete set of inputs for the direct ROI model does not exist until the project is complete. Thus, the direct ROI model provides one measure of value added for complete projects, but does not (except reasoning by analogy) help in determining the potential value added of candidate IV&V projects. A predictive ROI model will permit assessment of ROI of candidate projects and therefore assist managers in resource allocation. A predictive model can also serve as the basis for a model-based effectiveness metric that will permit progressive monitoring of ongoing IV&V projects.

A predictive ROI model based on the direct ROI methodology must provide the means to determine early in the project all inputs to the direct ROI model. Estimates of developer and IV&V cost should be available early in the lifecycle as these estimates are normal project management requirements. Other inputs, specifically developer and IV&V defect discovery data, can't be known early in the project and must therefore be estimated using project characteristics that are known early in the lifecycle.

Numerous techniques for software defect prediction have been proposed. Among these are regression-based approaches such as COCOQUALMO [4], capture-recapture models [7], and BBNs [8], [9]. Regression-based approaches have merit, but require far more test cases than were available for this work. Capture-recapture methods aren't amenable to this prediction problem because they estimate defect density after the analysis is complete. The BBN approach, however, is well suited to this problem, and good results have been achieved for similar problems [8], [9].

#### 3.1 BBN Overview

A BBN consists of a hierarchy of nodes representing stochastic causal relationships. Figure 2 depicts a single node with two inputs. The node output is a random value with a probability density function (pdf) which depends on the values of input parameters A and B. The mapping of parameter values to pdfs could be done using historical data, if sufficient data existed. In the absence of a sufficient quantity of data, the mapping can be estimated using expert opinion. For the ROI BBN, the expert opinion approach was selected. For the ROI BBN, all nodes produce random variables in the range of 1 to 5, where 1

corresponds to the worst possible case and 5 corresponds to the best possible case.

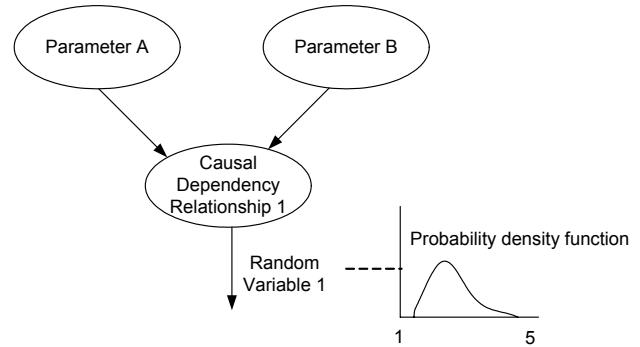


Figure 2: BBN Node

A complete BBN consists of a hierarchical set of nodes. Figure 3 shows a larger BBN fragment containing three nodes. Note that the inputs to node 3 are random variables and therefore node 3 must compute an expected probability density function. In practice, this is easily accomplished using the Monte Carlo method.

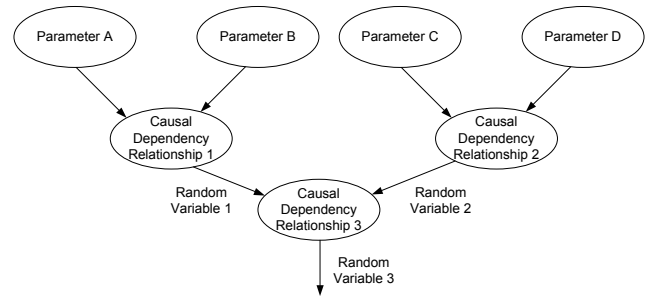


Figure 3: Hierarchical BBN nodes

#### 3.2 Function Point Ratios

For each development phase (requirements, design, code, test, integration), three BBN subnets were developed. The first subnet estimates product quality (with respect to defect density) on a scale of 1 to 5. The second subnet estimates developer defect detection efficiency on a scale of 1 to 5. The third subnet estimates IV&V defect detection efficiency on a scale of 1 to 5. In order to produce the inputs required by the direct ROI model, the BBN output must be converted into measures of defect size. Function points are a convenient measure early in the lifecycle because they can be estimated from system requirements and are independent of programming language. Therefore the subnet values (product quality, developer defect discovery efficiency, IV&V defect discovery efficiency) are used to estimate function point ratios (FPRs), which, scaled by the estimated product function points, provide suitable direct ROI model inputs. A high level flow diagram for the process (showing requirements phase defect function points only) is shown in Figure 4.

The overall process of developing the BBN model for one phase (requirements, design, code, test, integration) of the baseline IV&V process consists of the following steps:

1. Develop pictorially (based on elicitation from experts) the BBNs for defect introduction, defect detection by the developer, and defect detection by IV&V.
2. Elicit from the experts probability density functions for each causal dependency.

3. Implement the BBN in software using the Monte Carlo technique.
4. Calibrate the BBN output to case study data to predict discovered defect function points in-phase for the developer and IV&V

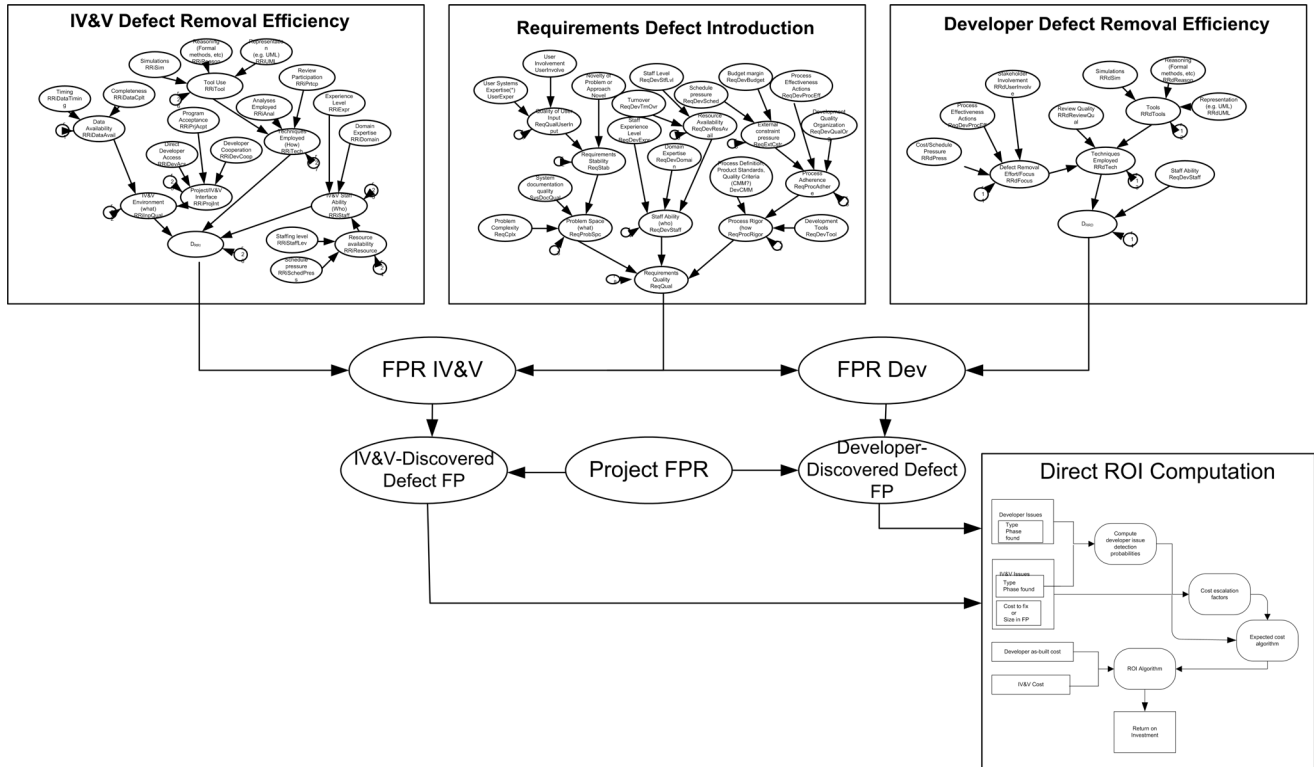


Figure 4: Predictive model defect function point computation (requirements phase)

### 3.3 Defect Leakage Model

The direct ROI model requires as inputs discovered-defect function points for the developer and IV&V for each issue type and phase found. The BBNs were developed to compute in-phase-discovered defects only. Although it would be possible to develop a BBN for each issue type for each development phase, that would require an additional fifteen subnets many of which would lack calibration data. Therefore, a leakage model was devised to estimate discovered defect function points out of phase (for example, requirements defect function points discovered in design, code, test, integration phases). Based on an extensive literature search, the Rayleigh leakage model [10], [11] was selected. Calibration of the leakage model will be discussed in a future paper.

### 4 Model Calibration

After eliciting pdf functions, it was necessary to calibrate the predictive model to the case study data. The calibration was performed in four steps. First, BBN input data was collected from project managers for four direct ROI case studies. FPRs were calculated for in-phase and leakage issues using the data collected for the direct ROI case studies. Next, FPR functions were developed for in-phase issues for each defect type (requirements, design, code, test, integration) for developer and IV&V issues. Last, the leakage model was calibrated to predict developer and IV&V defect detection in subsequent life cycle phases. Using the calibrated model, ROI was predicted for each of the four case studies and compared to case study results.

#### 4.1 BBN Input Data Collection

The prototype predictive model was calibrated using four direct ROI case studies [3]. Inputs for each BBN subnet for each of the four case studies were collected from IV&V project managers familiar with the case study projects. The inputs were collected using spreadsheets with embedded instructions. The spreadsheets included internal nodes to aid in validating the BBN topology and to ensure consistent data. Inconsistencies between internal nodes were discussed with the project managers and adjustments made to achieve consistent input. The score for each node is an estimated value in the range 1 - 10, a lower tolerance, and an upper tolerance. The range 1 - 10 was chosen because preliminary experiments indicated that eliciting inputs in the range 1 - 5 provided insufficient discrimination among projects. The inputs are implemented as triangular probability density functions as illustrated in Figure 5 and mapped linearly to a 1 - 5 range for compatibility with the BBN interpolation functions.

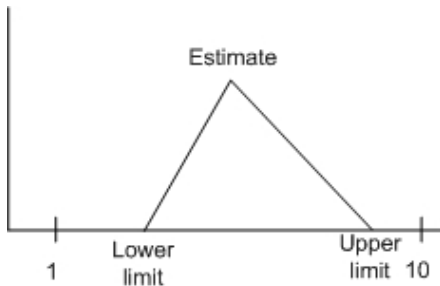


Figure 5: Node input pdf

#### 4.2 FPR Calibration

The FPR calibration mapped the three BBN outputs (quality  $Q_\phi$ , developer defect discovery efficiency  $D_{\phi\phi d}$  and IV&V defect removal efficiency  $D_{\phi\phi i}$ , where  $\phi$  represents the development phase and issue type). The mapping functions consist of lookup tables for each phase for developer and IV&V that map quality and efficiency to FPR. The main steps in the calibration for each issue type were as follows:

- Run the BBN model for each case study using the elicited BBN inputs
- Convert the case study defect data (actual data) to FPR
- Plot the actual FPR as a function of quality and defect discovery efficiency to identify candidate approximating functions
- Fit approximating functions to the BBN output and case study FPR
- Generate FPR lookup tables suitable for cubic spline interpolation and implementation in the BBN models
- Re-run the BBN using the calibrated FPR functions and compare computed FPR with actual FPR.

#### 4.2.1 Developer-Discovered FPR Calibration

For developer-discovered issues, plotting actual FPR vs requirements quality and developer defect removal efficiency, it was observed that FPR varies directly with the distance from the origin in the  $(Q_R, D_{RRd})$  plane. Therefore, an approximating function of the form

$$FPR_{RRd} = c_{RRd} \sqrt{Q_R^2 + D_{RRd}^2}$$

was used. The calibrated curve is shown in Figure 6. Here the circles are the case study data points and the solid trace is the approximating function. The corresponding interpolating surface is shown in Figure 7.

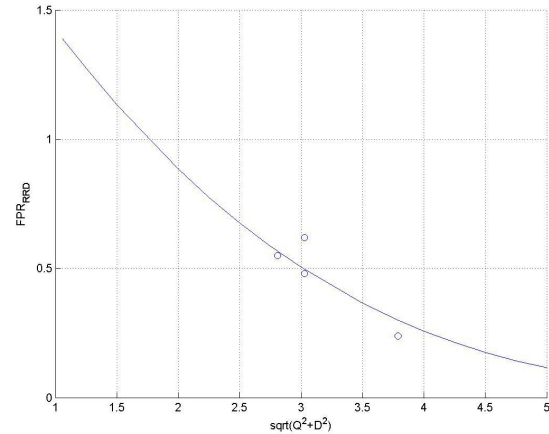


Figure 6: Requirements phase  $FPR_{DDd}$  calibration

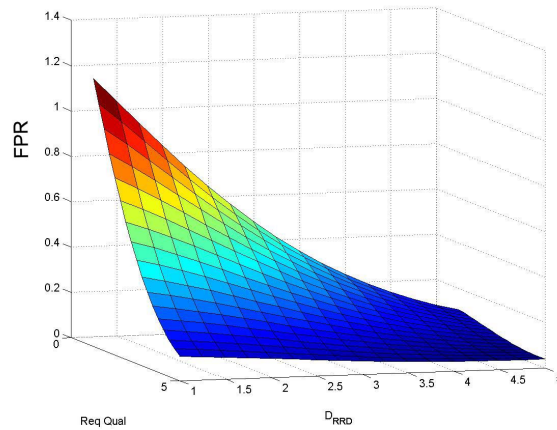


Figure 7:  $FPR_{RRd}$  interpolating surface

#### 4.2.2 IV&V-Discovered FPR Calibration

For IV&V-discovered issues, a more complex relationship was postulated and observed to fit well to the four case studies. The defect detection opportunity was postulated to be a function of  $Q_\phi$  such that there is an optimal point that depends on issue type. For  $Q_\phi$  values lower than the optimal point, the defect discovery opportunity is reduced because defect discovery is

inherently more difficult. For  $Q_\phi$  values greater than the optimal point, defect discovery opportunity is reduced because there should be fewer defects to discover. It was also observed that the effectiveness of IV&V defect discovery exhibits an inverse exponential relationship with IV&V efficiency score. The best fit for IV&V effectiveness was a function of the form

$$Eff_{IV\&V} = a e^{-bD_{\phi\phi i}} \cos(c D_{\phi\phi i})$$

where  $a$ ,  $b$ ,  $c$  are coefficients determined using a nonlinear least squares technique. Due to the difficulty in fitting functions to the opportunity data, interpolating tables were produced graphically and used to generate interpolating surfaces for FPR. An example IV&V FPR interpolating surface is shown in Figure 8.

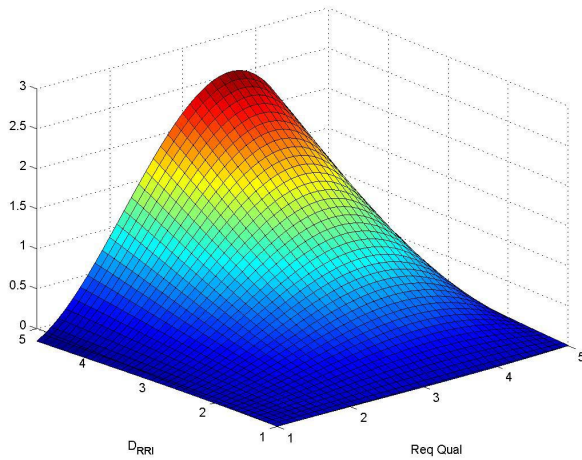


Figure 8: FPR<sub>RRI</sub> interpolating surface

#### 4.2.3 FPR Calibration Results

Good agreement was achieved between the measured FPRs and predicted values. As an example, Table 1 presents the FPR calibration results for the requirements phase. For the requirements phase, actual and predicted FPRs agree within one standard deviation. Similar results were achieved for the subsequent lifecycle phases, except for a few outliers due to project anomalies. For example, one project produced no formal design documentation.

### 5 Conclusions and Future Work

For the case studies completed to date, using the full-lifecycle BBN to predict FPR appears to be a reliable means to produce inputs to the IV&V direct ROI model. Additional case studies will be necessary to validate the overall model and refine the calibration.

### 6 Acknowledgements

This research was supported by the NASA Independent Verification and Validation Center and L-3 Titan Group.

Table 1: Requirements phase FPR calibration

Case	Developer FPR			IV&V FPR		
	Actual	Model	Std Dev	Actual	Model	Std Dev
A	0.551	0.709	0.166	0.096	0.135	0.098
B	0.483	0.629	0.176	0.079	0.093	0.070
C	0.239	0.371	0.135	0.812	0.603	0.286
D	0.623	0.628	0.161	0.137	0.289	0.169

### 7 References

- [1] J. Dabney, G. Barber, and D. Ohi, "Estimating direct return on investment of independent verification and validation," The 8<sup>th</sup> International Conference on Software Engineering and Applications, Cambridge, MA, November, 2004.
- [2] *Guidelines and Discount Rates for Benefit-Cost Analysis of Federal Programs*, Circular A-94, Office of Management and Budget, Washington, D.C., 1992.
- [3] J. B. Dabney, G. Barber, and D. Ohi, "Return on investment of independent verification and validation: Indirect benefits," 2003 Software Assurance Symposium, Morgantown, WV, 2003.
- [4] B. Boehm, C. Abts, A. W. Brown, S. Chulani, B. Clark, E. Horowitz, R. Madachy, D. Reifer, and B. Steece, *Software Cost Estimation with COCOMO II*, Prentice Hall, Upper Saddle River, NJ, 2000.
- [5] J. B. Dabney and G. Barber, "Direct Return on Investment of Software Independent Verification and Validation: Methodology and Initial Case Studies," 2003 Assurance Technology Symposium, Cleveland, OH, 2003.
- [6] *Function Point Counting Practices Manual*, Release 4.1.1, The International Function Point User's Group, 2000.
- [7] L. C. Briand, K. El Emam, B. G. Freimut, O. Laitenberger, "A comprehensive evaluation of capture-recapture models for estimating software defect content," IEEE Transactions on Software Engineering, Vol 26, No. 6, June, 2000, pp. 518 – 540.
- [8] N. Fenton, P. Krause, and M. Neil, "Software measurement: Uncertainty and causal modeling," 2001.
- [9] N. Fenton, P. Krause, M. Neil, "A probabilistic model for software defect prediction," preprint, University of London, England, 2001.
- [10] J. W. E. Greene, "Purchasing Software Intensive Systems Using Quality Targets," Quality Software Management Ltd., 2001
- [11] S. H Kan, "Modeling And Software Development Quality," IBM Systems Journal, Vol 30, No. 3, 1991