A Process to Calculate the Uncertainty of Software Metrics-based Models Using Bayesian Networks

Renata M. Saraiva*[1], Mirko Perkusich†[1], Hyggo Almeida‡[1], and Angelo Perkusich§[1]

[1]Embedded and Pervasive Computing Laboratory, Federal University of Campina Grande, Campina Grande, Brazil

Abstract

Software metrics are essential resources in software enterprises. They can be used to support decision-making and, consequently, reduce costs, improve the productivity of the team and the quality of products delivered. On the other hand, this is only possible if the metrics are valid. Although there are studies related to software metrics validity, none present a solution to represent the uncertainties of the metrics selected to measure the attributes of the entities. In this paper, we present a process to build Bayesian networks to represent the uncertainties of software metrics-based models. The proposed solution is composed of two activities and focuses on the selection and validation of metrics to construct the Bayesian networks. We validated the model with simulated scenarios. Given the successful results, we concluded that the proposed solution is promising. This paper complements the state of the art by showing how to complement a popular metric selection technique, GQM, with information to model uncertainties of the metrics using the concepts of metric validation and Bayesian networks.

Software metrics selection; Software metrics validation; Goal-Question-Metric; Validation criteria; Bayesian networks.

1 Introduction

According to Finkelstein and Leaning [9], measurement is the objective representation of an empirical knowledge of a real-world entity. According to Mathias et al [13], a measurement occurs when an attribute is measured, that is, when a value is assigned to it. By combining this measure with useful information, we have a metric (e.g., average number of defects per module). In practice, the terms "metric" and "measure" are often used interchangeably [13]. In this paper, we use the definitions presented by Mathias et al. [13]. However, we consider that each measure is a metric and this, in turn, may be composed of more than one measures.

There are many applications in the field of software metrics such as quality assessment and prediction. For instance, Quamoco [19] focuses on measuring product quality and Hearty et al. [11], on predicting the velocity of an XP project.

Despite the benefits of using metrics and the various researches on software metrics undertaken in recent years, the acceptance and use of metrics in practice is still an ongoing concern: more than 80% of software measurement initiatives fail within the first 18 months. One possible explanation for this phenomenon is the difficulty to understand and use metrics [20].

According to Fenton and Neil [7], metrics have been used successfully to quantify, but they have not been properly used to support decision-making. A reason for the limited adoption with this purpose is the lack of trustworthiness on the validity of metrics. For instance, Chidamber-Kemerer (CK) metrics are popularly used to evaluate Object Oriented-based software. On the other hand, Kitchenham [12] discusses that two of the proposed metrics (Lack of Cohesion and Coupling Between Objects) are theoretically invalid. In other words, they do not represent the attributes of the entities in which they were proposed to. Using invalid metrics result in meaningless (i.e., totally arbitrary) decisions.

There are studies that propose criteria to evaluate the validity of software metrics. In Meneely et al. [14], results

*renata.saraiva@embedded.ufcg.edu.br
†mirko.perkusich@embedded.ufcg.edu.br
‡hyggo@embedded.ufcg.edu.br
§perkusic@embedded.ufcg.edu.br

DOI reference number: 10.18293/SEKE2017-172

of a systematic literature review are presented, in which 47 criteria were identified. On the other hand, there are no proposed solutions to, given an attribute of an entity that needs to be measured, calculate how representative (i.e., valid) is the set of metrics selected to measure it. In the context of Goal Question Metric (GQM), a popular software metrics paradigm, there is no solution to model the uncertainty of the set of metrics used to answer a question using the criteria presented in Meneely et al. [14].

In this paper, we present a process to build Bayesian networks to represent the uncertainties of software metrics-based models. The process is composed of two activities: (i) metrics selection and (ii) metrics validation. The first activity is composed of three steps: characterization of the environment, acquisition of knowledge through abstraction sheets and construction of the Bayesian network. The second activity is composed of two steps: execution of the validation method from validation criteria and update the Bayesian network.

We used Bayesian networks because they are flexible to be learned from data or elicited from domain experts. Since metric models can be applied in contexts in which there are historical data and in which there are not, this flexibility is crucial. Furthermore, it can deal with different types of data (e.g., discrete, continuous, Boolean and ordinal), which adds flexibility to the types of metrics to be used. Finally, it deals with uncertainty and enables the modeling of cause-consequence relationships, which enables the modeling of the validity of metrics and build GQM-based models.

To validate our solution, we used ten simulated scenarios. Based on the results, we concluded that it is a promising approach to assist on the construction of interpretation-oriented metric programs. We plan to complement our process with threshold definition techniques [17] and metrics reliability activities [15]. This paper complements the state of the art by showing how to complement a popular metric selection technique, GQM, with information to model uncertainties of the metrics using the concepts of metric validation and Bayesian networks.

This paper is organized as follows. Section 2 presents an overview on Bayesian networks. Section 3 presents our proposed solution. Section 4 presents our validation and Section 5 presents our final remarks.

2 Bayesian Networks

Bayesian networks are probabilistic graph models used to represent knowledge about an uncertain domain [2]. A *Bayesian network*, N, is a directed acyclic graph that represents a joint probability distribution over a set of random variables V [10]. The network is defined by the pair $N = \{G, \Theta\}$. G is the directed acyclic graph in which the nodes $X_1, \ldots, X_n$ represent random variables and the arcs represent the direct dependencies between these variables. Θ represents the set of the probability functions. This set contains the parameter $\theta_{x_i|\pi_i} = P_N(x_i|\pi_i)$ for each x_i in X_i conditioned by π_i, the set of the parameters of X_i in G. Equation 1 presents the joint distribution defined by N over V.

$$P_N(X_1, \ldots, X_n) = \prod_{i=1}^{n} P_N(x_i|\pi_i) = \prod_{i=1}^{n} \theta_{X_i|\pi i} \qquad (1)$$

Bayesian networks have many advantages such as suitability for small and incomplete data sets, structural learning possibility, combination of different sources of knowledge, explicit treatment of uncertainty, support for decision analysis, and fast responses [18]. Furthermore, they can combine the knowledge of domain experts and historical data to build more realistic models in an approach called smart-data [4]. To construct the Bayesian networks presented in this study we used AgenaRisk[1].

This technique has been applied to build software metrics-based models for several purposes in software engineering such as risk management [6], product quality management [19], effort prediction [11] and process management [15].

To reduce the effort of defining the Node Probability Tables (NPTs) through elicitation of knowledge from domain experts, Fenton et al. [8] proposed the concept of ranked nodes, which is based on the doubly truncated Normal distribution (TNormal) limited in the $[0, 1]$ region. We used ranked nodes because the goal is to give meaning to the metric. Therefore, we used an ordinal scale. An advantage of ranked nodes, when compared to other approaches to define NPT for ordinal variables, is the explicit configuration of the confidence in the result (i.e., variance).

3 Proposed Process

The goal of the proposed solution is to represent the uncertainties of software metrics-based models. For this purpose, we used Bayesian networks. It is composed of two activities: (i) metrics selection and (ii) metrics validation. The first activity is composed of three steps: characterization of the environment, acquisition of knowledge through abstraction sheets and construction of the Bayesian network. The second activity is composed of two steps: execution of the validation method from validation criteria and update the Bayesian network. In Figure 1, we present an activity diagram representing the process.

3.1 Software Metrics Selection

To select the metrics, we use the GQM [1] paradigm. First, it is necessary to identify the project context (i.e.,

[1] http://www.agenarisk.com/

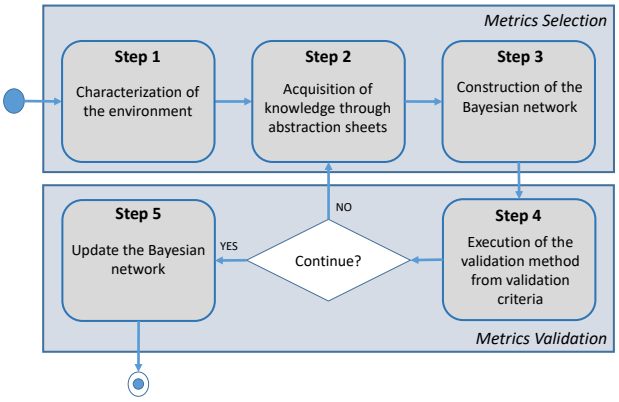

Figure 1. Process overview.

application domain and development process) from the domain experts. For instance, semi-structured interviews with project leaders might be performed.

With the context defined, the next step is to apply the GQM process with the goal of capturing the experience of the viewpoints and selecting the software metrics. For this purpose, abstraction sheets must be used as knowledge acquisition instrument during interviews [3].

A set of goals is defined as $G = \{g_1, \ldots, g_{|G|}\}$, where g_i represents a project goal. For each goal, an abstraction sheet, which is composed of four quadrants, must be used. A set of questions, Q, and metrics, M, must be extracted from the first quadrant, which corresponds to the quality focus. $q_i \in Q$ and $m_i \in M$ represent, respectively, a question and a metric related to a project goal.

The second quadrant corresponds to the variation factors, which are factors that impact the quality focus, considering the defined goal. More questions q and metrics m can be extracted from this quadrant.

The third quadrant corresponds to the possible values of the extracted metrics from the first quadrant. These values are important because they demonstrate the usefulness of the measurement process. By analyzing them, we can detect discrepancies between expectations and realities.

Finally, the last quadrant of the abstraction sheet corresponds to the impact of the variation factors on the extracted metrics from the first quadrant. The description of this impact serves as motivation for the inclusion of the variation factor in the abstraction sheet. If the project leader does not know how to inform the impact of a variation factor, this factor should be excluded from the sheet.

Given that each question must be associated with at least one goal, a set of relationships between goals and questions, T, should be created. $t(g, q)$ means the goal g and question q are related. An hierarchical structure example of the GQM model is illustrated in Figure 2.

For the example shown in Figure 2, the given goal is to "Analyze the software product with respect to its quality for the purpose of characterization from the developer's point of view". For quality focus, the question "How many unwanted behaviors does the product have?" was defined. For the variation factor, the question "What is the quality of the test?" was defined. An example of an abstraction sheet is illustrated in Figure 3.

By analyzing first quadrant of the abstraction sheet presented in Figure 3, it is possible to identify metrics such as *number of detected failures*, *proportion of critical/uncritical failures* and *number of detected faults*. Given that the goal, questions and metrics for the quality of focus are defined, the Bayesian network can be built. For our approach, all node should be modelled as ranked. If a metric is collected using a numerical scale, thresholds must be defined to convert it into an ordinal scale. For this purpose, statistics-based approach [17] can be used or data must be collected from domain experts.

After the construction of the directed acyclic graph, the NPT of the goal node must be defined. Assuming that the goal was modeled as a ranked node, we can create a truth table to collect data from a domain expert and define the NPT. For the given example, given that there is a one-to-one relationship between g and q, the truth table is not necessary and the NPT must be calibrated as an identity matrix, in which the diagonal elements are 1 and the remaining are 0.

3.2 Software Metrics Validation

The validation of the metrics ensures that they are representative of the measured attributes. In the literature, there are many researches on the validation of software metrics [16, 12, 14]. Meneely et al. [14] performed a systematic review about validation criteria for software metrics and identified 47 criteria. In Table 1, we present ten criteria identified by Meneely et al. [14].

A priori validity	*Monotonicity*
Actionability	*Metric Reliability*
Appropriate Continuity	*Non-collinearity*
Appropriate Granularity	*Non-exploitability*
Association	*Non-uniformity*

Table 1. List of 10 validation criteria found in the review [14].

Defining the purpose of using a metric is a critical step to validate it. In addition, when the project leader makes a decision, he can specify properties of the metrics that are most appropriate to use. According to Meneely et al. [14] this is called advantage. For instance, be able to show that a metric is a significant representation and that it can be applied to a development process are considered advantages.

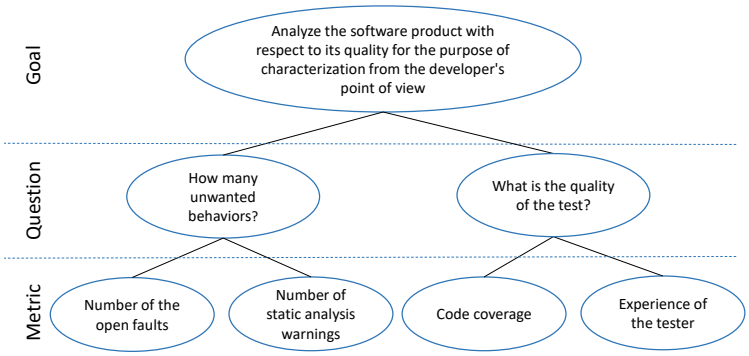

Figure 2. Example of a GQM model.

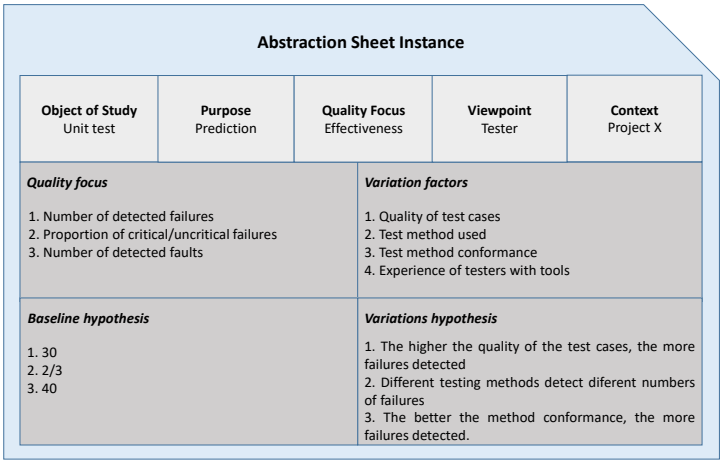

Figure 3. Abstraction sheet instance.

To execute this activity, we consider the validation criteria presented in Meneely et al. [14] due to its completeness. For each metric m related to the quality focus, the following steps must be performed:

1. Determine the intended use of the metric.
2. Highlight the advantages that are appropriate for the intended use chosen in the previous step. Some of these advantages are: mathematical soundness, practicality, correctness, efficiency and hypothesis-strengthening [14].
3. Look up the validation criteria that are tied to the advantages shown in Table 2.
4. Carefully choose validation criteria while considering the purpose of the metrics and the relationships and motivations among the criteria.
5. Analyze if the metric follows the chosen validation criteria.

#	Criterion	*Mathematical Soundness*	*Practicality*	*Correctness*	*Efficiency*	*Hypothesis-Strengthening*
1	*A Priori Validity*					X
2	*Actionability*		X			
3	*Appropriate Continuity*	X				
4	*Appropriate Granularity*					
5	*Association*					

Table 2. Example of mapping from criteria to advantages [14].

Given the concept of *variance* of ranked nodes, we can model the confidence in the validity of a set of metrics defined to answer a question. The greater the confidence that a set of metrics is valid to represent an attribute, the smaller that variance. To define the variance, the given rules of thumb should be used:

Rule 1: If the metric follows 100% of the validation criteria related to it, the variance must be equal to $5x10^-4$, the smallest value possible in AgenaRisk;

Rule 2: If the metric follows between 50% and 99% of the validation criteria related to it, the variance must be equal to $5x10^-3$;

Rule 3: If the metric follows between 1% and 49% of the validation criteria related to it, the variance must be equal to $5x10^-2$;

Rule 4: If the metric does not follow any validation criteria related to it, the variance must be equal to $5x10^-1$;

These intervals are recommendations for the first calibration of the Bayesian network and are restricted to AgenaRisk, which is currently the only software that supports ranked nodes. Another approach is to use the validation criteria as a reference and elicit knowledge from the domain expert to calibrate the variance. Furthermore, it is possible to, after applying the model, refine the calibration of the NPTs given knowledge from experts and collected data.

If a given node q has more than one parent node (i.e., more than one metric), the described rules should be applied considering the sum of the validation criteria for each metric.

The next step is to finalize the calibration of the NPT by, as presented in [8], defining a function to model the central tendency of the distribution that represents the NPT. To define the functions, weights and variance, knowledge must be elicited from the experts using the approach presented by Fenton et al. [8] or da Silva et al. [5].

4 Validation

We validated the resulting Bayesian networks in ten simulated scenarios. For all cases, we assumed that the first step of the proposed process was successfully executed. Due to space limitations, we only present the results of one scenario. This scenario describes a simple product quality model, where the goal is "Effectiveness of unit test', the question is "How many unwanted behaviors does the product have?' and the metrics are *number of detected failures* and *number of detected faults*.

In this case, given that the goal of using the metrics *number of opened faults* and *number of static analysis warnings* is to assist on decision-making during the development of the software, the advantage highlighted is *Decision-Informing*. There are 11 validation criteria associated with this advantage. On the other hand, say that the metric *number of opened faults* conforms to 8 out of 11 criteria and the metric *number of static analysis warnings* conforms to 5. Given this, together, both metrics conform to 13 out 22 validation criteria (i.e., 59.09%). Therefore, the second rule will be followed and the variance of the node *Unwanted behaviors* is $5x10^-3$.

To calibrate the NPT, we used the WMIN function, because if any of the given metrics are *Very low*, the answer to the corresponding question will tend to *Very low*. On the other hand, we considered *number of opened faults* as more important than *number of static analysis warnings*. Therefore, we defined them, respectively, with weights 2 and 1. We show an example of the calculated results for this scenario in Figure 4.

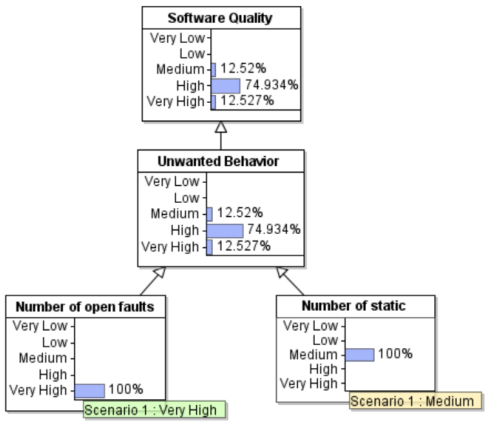

Figure 4. Example of a Bayesian network constructed using the proposed solution.

By analyzing Figure 4, it is possible to notice that, given the validity of the metrics used to answer the question, the confidence in the decision regarding the goal is acceptable. In case it was not, the probability of the goal would tend to be more uniform, meaning that a reliable decision could not be taken given the constructed model.

5 Final Remarks

In this paper, we presented a process to build Bayesian networks to represent the uncertainties of software metrics-based models. The process is composed of two activities: (i) metrics selection and (ii) metrics validation. The first activity is composed of three steps: characterization of the environment, acquisition of knowledge through abstraction sheets and construction of the Bayesian network. The second activity is composed of two steps: execution of the validation method from validation criteria and update the Bayesian network.

The process shown is based on the concept of ranked nodes [8] to build the Bayesian networks, with the goal of

adding meaning to metrics. Furthermore, it uses GQM to assist on the selection of metrics and software metrics validation criteria extracted from Meneely et al. [14]. On the other hand, if necessary, other types of nodes can be used such as Boolean, but it will be necessary to define a new reasoning to map the validity and the NPT definition.

The main limitation is the study's validation, which is only conceptual. On future works, we will execute empirical studies to evaluate our approach by collecting data from practitioners and tools to assess if the proposed solution improves the accuracy of decision making. Furthermore, we will complement our solution with additional steps regarding the definition of software metrics thresholds and collection reliability to assist on the construction of interpretation-oriented software metrics models.

References

[1] V. R. Basili. Software modeling and measurement: The goal/question/metric paradigm. Technical report, College Park, MD, USA, 1992.

[2] I. Ben-Gal. *Bayesian Networks*. John Wiley and Sons, 2007.

[3] L. C. Briand, C. M. Differding, and H. D. Rombach. Practical guidelines for measurement-based process improvement. *Software Process Improvement and Practice*, 2(4):253–280, 1996.

[4] A. Constantinou and N. Fenton. Towards smart-data: Improving predictive accuracy in long-term football team performance. *Knowledge-Based Systems*, pages –, 2017.

[5] R. da Silva, M. Perkusich, R. Saraiva, A. Freire, H. Almeida, and A. Perkusich. Improving the applicability of bayesian networks through production rules. In *27th International Conference on Software Engineering and Knowledge Engineering*, SEKE 2016, page In press, San Francisco, USA, 2016.

[6] C.-F. Fan and Y.-C. Yu. Bbn-based software project risk management. *Journal of Systems and Software*, 73(2):193–203, Oct. 2004.

[7] N. E. Fenton and M. Neil. Software metrics: roadmap. In *Proceedings of the Conference on the Future of Software Engineering*, pages 357–370. ACM, 2000.

[8] N. E. Fenton, M. Neil, and J. G. Caballero. Using ranked nodes to model qualitative judgments in bayesian networks. *IEEE Trans. on Knowl. and Data Eng.*, 19(10):1420–1432, Oct. 2007.

[9] L. Finkelstein and M. Leaning. A review of the fundamental concepts of measurement. *Measurement*, 2(1):25–34, 1984.

[10] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, 1997.

[11] P. Hearty, N. Fenton, D. Marquez, and M. Neil. Predicting project velocity in xp using a learning dynamic bayesian network model. *Software Engineering, IEEE Transactions on*, 35(1):124–137, 2009.

[12] B. Kitchenham. Whats up with software metrics?–a preliminary mapping study. *Journal of systems and software*, 83(1):37–51, 2010.

[13] K. S. Mathias, J. H. Cross II, T. D. Hendrix, and L. A. Barowski. The role of software measures and metrics in studies of program comprehension. In *Proceedings of the 37th annual Southeast regional conference (CD-ROM)*, page 13. ACM, 1999.

[14] A. Meneely, B. Smith, and L. Williams. Validating software metrics: A spectrum of philosophies. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 21(4):24, 2012.

[15] M. Perkusich, A. Medeiros, K. C. Gorgônio, H. O. de Almeida, A. Perkusich, et al. A bayesian network approach to assist on the interpretation of software metrics. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing*, pages 1498–1503. ACM, 2015.

[16] N. F. Schneidewind. Methodology for validating software metrics. *IEEE Transactions on software engineering*, 18(5):410–422, 1992.

[17] R. Shatnawi. Deriving metrics thresholds using log transformation. *Journal of Software: Evolution and Process*, 27(2):95–113, 2015.

[18] L. Uusitalo. Advantages and challenges of bayesian networks in environmental modelling. *Ecological Modelling*, 203(34):312 – 318, 2007.

[19] S. Wagner, A. Goeb, L. Heinemann, M. Kls, C. Lampasona, K. Lochmann, A. Mayr, R. Plsch, A. Seidl, J. Streit, and A. Trendowicz. Operationalised product quality models and assessment: The quamoco approach. *Information and Software Technology*, 62:101 – 123, 2015.

[20] L. G. Wallace and S. D. Sheetz. The adoption of software measures: A technology acceptance model (tam) perspective. *Information & Management*, 51(2):249–259, 2014.