Key words:
software risk assessment, risk framework,
Bayesian nets, project trade-offs

Łukasz RADLIŃSKI*
Norman FENTON**

CAUSAL RISK FRAMEWORK FOR SOFTWARE PROJECTS

A traditional approach to quantitative risk assessment is to calculate the "risk score" of an event as the product of its "probability" and "impact". But the "probability" and "impact" of an event cannot meaningfully be evaluated without incorporating factors that could prevent an event from occurring or to limit the negative consequences if it does occur. The "Risk score" is often interpreted as expected financial loss. Project managers are often interested in estimating other types of consequences such as: missing the deadline, need for additional staff, lower quality of delivered product etc. Bayesian nets (BNs) have the potential to extend traditionally perceived risk assessment by adding several features such as causality, uncertainty, mixture of empirical data with expert knowledge, and intuitiveness. In this paper we propose a causal risk framework (CRF) for software projects, which significantly extends traditional risk assessment. The CRF is based on risk categorisation depending on user perspective (role in the project). We use five components of risk: "trigger", "control", "risk event", "mitigant", "consequence". Analysis of existing BNs built for the software engineering area reveals that the potential use for most of them is limited because they are not directly compatible with CRF. We discuss the benefits of applying CRF to a Software Project Trade-off Model and directions for future research in applying CRF to software projects.

1. INTRODUCTION

Project risk management is one of nine knowledge areas defined in the Project Management Body of Knowledge [17]. With an observed increase of project complexity risk

---

* Department of Information Systems Engineering, University of Szczecin, ul. Mickiewicza 64, 71-101 Szczecin, Poland

** Department of Computer Science, Queen Mary, University of London, Mile End Road, London E1 4NS, United Kingdom

management becomes more important. One of the stages in risk management is risk assessment for which many techniques have been proposed. Although these techniques provide some information for decision makers they are limited by the lack of quantitative integration of both controllable and uncontrollable factors. To extend and improve the quantitative risk assessment a Causal Risk Framework (CRF) based on Bayesian nets (BNs) has been proposed in [9], [10].

In this chapter we discuss the need for CRF integrated with BNs and show possible extensions which may be useful in more complex models. We analyze the applicability of CRF to existing BNs built for the software engineering area and focus on reasons why many of the existing models are not easily compatible with CRF. We discuss in detail how CRF can be integrated with a Software Project Trade-off Model – a BN that initially was not built with the requirement to be compatible with CRF but which was relatively easy restructured to meet the criteria of CRF.

2. QUANTITATIVE RISK ASSESSMENT

There are two common approaches in defining risk. The first approach highlights the negative aspect of risk, e.g. in definition: "possibility of loss or injury", "someone or something that creates or suggests a hazard" [14]. The second approach assumes both the positive and negative aspects of risk, e.g.: "an uncertain event or condition that, if it occurs, has a positive or negative effect on at least one project objective" [17], "an uncertain event or set of circumstances that, should it occur, will have an effect on the achievement of the project's objectives [21]. Our approach assumes both the positive and negative aspects of risk and extends the terminology by defining components of risk (discussed later in this chapter).

Quantitative risk assessment enables assessment of risk quantitatively using various techniques such as: sensitivity, expected monetary value, decision tree, Monte Carlo analyses. Quantitative risk assessment typically captures risk in terms of *risk exposure*:

$$\textit{risk exposure} = \textit{probability of event} * \textit{impact of event} \qquad (1)$$

There are three main practical problems associated with this definition [9]:

1. it is both difficult and meaningless to estimate the *probability of event* without analyzing various factors that determine the occurrence of the event and the possible actions undertaken to prevent the event from occurring;
2. it is also often difficult and meaningless to estimate the *impact of event* without analyzing factors that can limit the negative or extend the positive consequences;
3. *risk exposure* can be used in prioritizing risks (as done in qualitative risk assessment) but otherwise its interpretation as *potential loss* or more generally as *potential outcome* is not very useful.

3. CAUSAL RISK FRAMEWORK

To overcome the limitations discussed earlier we introduce a Causal Risk Framework which provides a perspective-dependent components of risk. The main motivations for using CRF are [9]:

– ensuring that every aspect of risk measurement is meaningful in its context,
– quantification of uncertainty through probability values associated with each component,
– providing a visual and formal mechanism for recording and testing subjective probabilities,
– simplifying the process of model building from the start.

There are two dimensions in CRF: **components of risk** and **perspectives**. Components describe a type of risk in a specific context while the perspectives describe this context. A perspective contains five risk components as initially proposed in [9], [10]:

– *risk event* – an event which is the focal point for a given perspectives,
– *trigger* – an event out of control for a given role which initiates or determines the occurrence of *risk event*,
– *control* – an action whose aim is to prevent *risk event* from happening or limit the scope of *risk event*,
– *consequence* – an event which reflects how *risk event* impacts on the wellness of an agent from a given perspective,
– *mitigant* – an action whose aim is to prevent negative *consequence* from happening or limit the extent of *consequence*.

Fig. 1 illustrates an example of CRF applied to a software project. All risk components are linked together according to the cause-effect relationships. *Requirements creep* is a risk event which is triggered by *poor requirements specification* and controlled by *constrained user expectations*. *Requirements creep* can be mitigated by *additional funding from customer* but this may lead to *project over budget*.

The industry-scale models are usually more complex than the example from Fig. 1. Typically there are more than one variable for each risk component, and there may be additional links between variables within a single risk component. The generic structure of such a model compatible with CRF is illustrated in Fig. 2. In addition to the five components of risk discussed earlier one more component can be added – *an indicator*. In many cases it is not possible or is too difficult to define the main five components quantitatively. On the other hand, some metrics describing these components numerically may be available. In such cases components of risk may be defined qualitatively, e.g. on a ranked scale, and numeric indicators can be added to these nodes for which numeric metrics and data exist. Such approaches using metrics as indicators have been previously used in [7], [24].

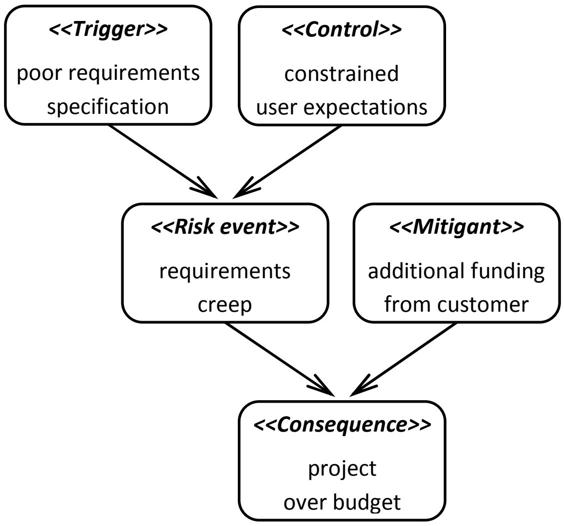

Fig. 1. Schematic of Causal Risk Framework in example software project [19]

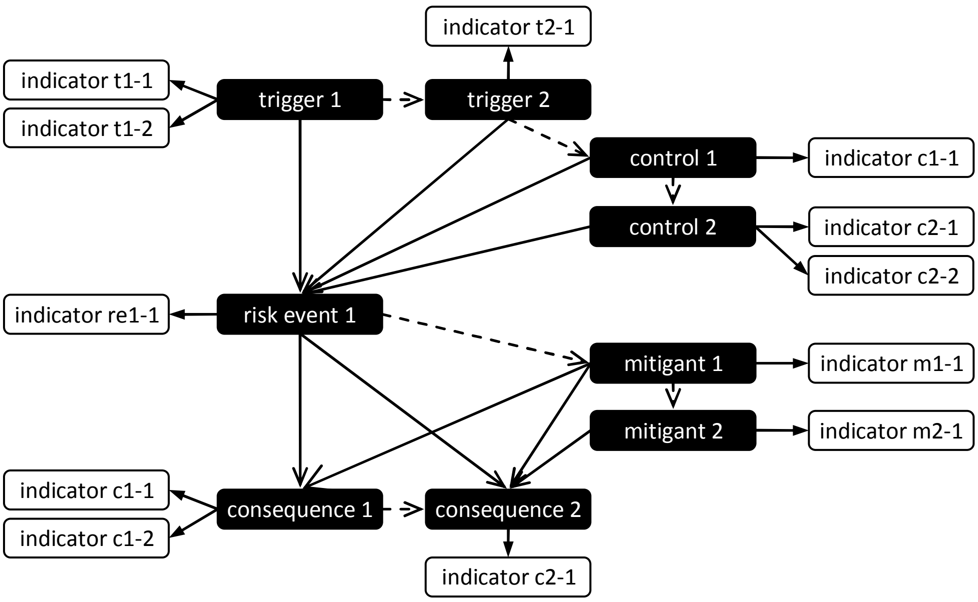

Fig. 2. Schematic of more complex Causal Risk Framework

Assigning model variables to risk components is not fixed. A variable may be a consequence from one perspective but a trigger or risk event from another perspective. Such examples are discussed later in this chapter. There is no unified list of perspectives because they depend on the system under consideration. It means that there may be different perspectives in software projects, medical diagnosis, public safety systems and other areas.

The basic CRF with two dimensions (components and perspectives) can be extended depending on user needs. For example, instead of five components of risk there may be a need for using more details, like distinguishing the levels of consequences: internal (current company or process state), end-product, future company states.

The second type of extension to CRF is adding a new dimension – **area**. An area describes the group of project variables to which a specific variable belongs to. These areas can either be standard or company-specific project management areas or a mixture of both. Table 1 contains a list of standard areas in software projects proposed in [23].

Table 1. Standard areas of software project management as third dimension of CRF

Mission and Goals	Project Parameters	Project Management
Program Management	Product Content	Project Team
Decision Drivers	Deployment	Technology
Organization Management	Development Process	Maintenance
Customer/User	Development Environment	

4. BAYESIAN NETS

4.1. BACKGROUND

Bayesian nets appears to be the technique which can be used in quantitative risk assessment compatible with CRF as it provides rich input for decision makers and overcomes limitations of traditional risk assessment techniques.

A Bayesian net can be perceived from two sides: *functional* – as a directed acyclic graph consisting of nodes representing random variables and arcs connecting pairs of nodes, and *analytical* – as set of unconditional and conditional probability tables defining variables and allowing rigorous inference based on well-established probability calculus, specifically using Bayes's theorem [2]. An example of a simple BN with all variables of Boolean type is illustrated in Fig. 3.

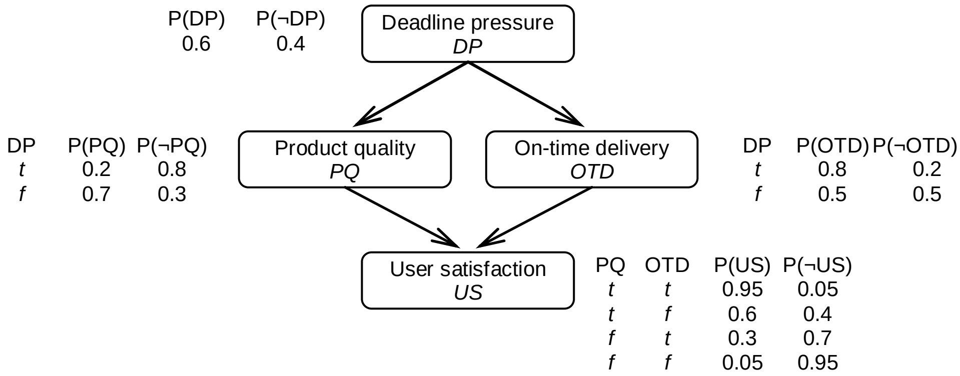

Fig. 3. Example of simple BN

The main advantages of using BNs in quantitative risk assessment are:

- capturing uncertainty explicitly,
- ability to build models based on statistical results, expert judgment or both,
- ability to combine qualitative and quantitative data,
- ability to run with incomplete data,
- ability to incorporate causal relationships,
- performing forward and backward inference.

4.2. APPLICATIONS IN SOFTWARE ENGINEERING

BNs have been applied in a number of studies in software engineering. Three main types of BN structures have been used:

1. *Naïve Bayesian Classifier* (NBC) which has a structure of a star and does not contain causal relationships between variables;
2. *Causal BN* which contains causal relationships between variables;
3. *Dynamic Bayesian Net* (DBN) which forms a series of sequentially linked causal BNs and thus reflects the dynamics of modeled process.

In general using BNs as a modelling technique does not ensure the possibility of applying CRF (Table 2). Lack of causal relationships in a model, i.e. in NBC, is the main obstacle in combining CRF with BNs. Even with the causal or DBN structure such combination may not be possible. When a BN has been developed with a structural learning method from a dataset the links between variables may not reflect cause-effect relationships but only statistical dependence. In some cases even if the model structure has been defined by expert(s) there are still missing causal links between some variables which are treated independently. This occurs in models focused on predicting a specific attribute, usually with forward only prediction, with little stress on reflecting all causal relationships. Another reason is the lack of controllable factors in some of models whose aim is to make prediction mostly on project (code) attributes. As a result of all these reasons many BNs that initially appear to be causal cannot be integrated with CRF or such integration would require serious change in the models' structure.

Table 2. Summary of recent BNs for software engineering

Ref.	Main problem analyzed	Type of BN	Compatibility with CRF
[5]	defect rate	causal	no: activities not linked according to causal relationships
[15]	maturity of requirements	causal	no: links mainly directly from input variables to dependent variable
[26]	testing process	causal	no: lack of controllable factors
[16]	fault content, fault proneness	causal	no: links only directly from input variables to dependent variable
[27]	change coupling	causal	no: no detailed information provided, structure learnt from data
[24]	various aspects of software quality	causal	partially: no detailed information provided, not all of risk components can be used
[4]	effectiveness of inspections	causal	unknown: limited detailed information provided
[7]	trade-off between: functionality, effort, quality	causal	yes
[18], [19]	trade-off between: functionality, effort, quality	causal	yes
[8], [11]	defects, partly: effort	causal/DBN	yes
[12]	project velocity (functionality)	DBN	no: lack of causal links between activities
[6], [19]	defects	DBN	partially: not all of risk components can be used
[3]	effort	DBN	partially: unclear impact of project and process factors
[1]	failures	DBN	unknown: very few causal links, limited detailed information provided
[22]	effort, productivity	NBC	no: lack of causal links

5. SOFTWARE PROJECT TRADE-OFF MODEL

The Software Project Trade-off Model appears to be one of the few analysed BNs to which CRF can be applied. The main feature of the SPTM is a trade-off analysis between key software project factors: development effort, functionality (size) and quality (number of defects and defect rate). For example, for the assumed functionality it is possible to analyze how much more effort is required to achieve better quality or how much lower quality should be expected with decreased amount of effort. All of these relationships incorporate the influence of project, process and people factors.

The schematic structure of SPTM is presented in Fig. 4. Each rectangle represents a set of variables that are connected according to cause-effect relationships. It is beyond the scope of this chapter to provide a detailed explanation for each variable (the model contains more than 50 variables), model behaviour and validation scenarios. These can be found in earlier work [6], [18], [19], [20]. Here we focus on applying CRF to SPTM. From this point of view we put two explanations on model structure: uncontrollable project factors aggregate: complexity, novelty, scale, quality of input documentation, deadline pressure, positive and negative customer involvement. SPTM does not contain design activity explicitly but through other activities: specification (design tasks related to producing documentation) and coding (design tasks related to producing code).

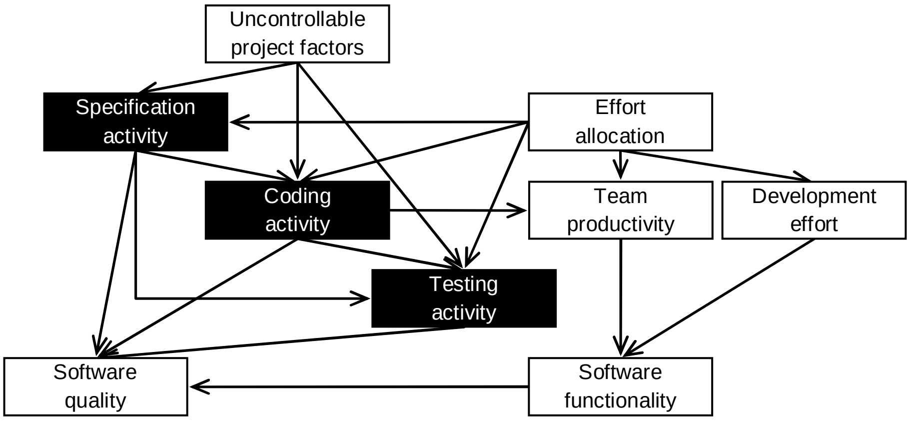

Fig. 4. Schematic of Software Project Trade-off Model

To make SPTM compatible with CRF it was necessary to restructure the model from previous editions. Currently SPTM supports three user perspectives: analytic, programmer and tester. Enabling more perspectives, e.g. project manager, is possible only after further deeper restructuring of the model.

Fig. 5. illustrates the risk components of CRF applied to SPTM from three perspectives. Most of the risk components contain more than one variable. To keep the graph clear variables within a single component have been shown as one graphical object. Several variables appear more than once. For example, *coding effectiveness* is a consequence in the analytic perspective, a risk event in the programmer perspective and a trigger in the tester perspective.

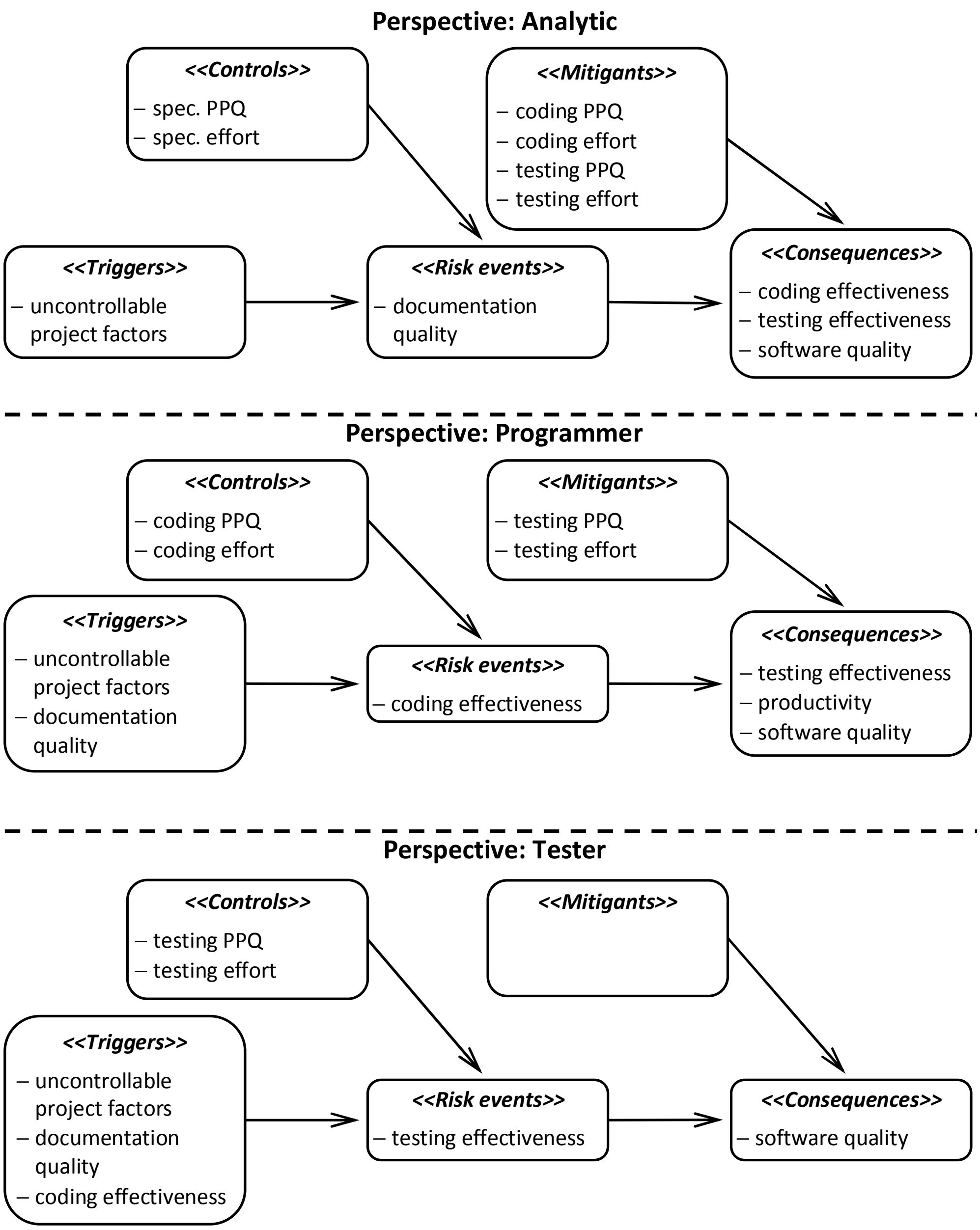

Fig. 5. Risk components for user perspectives in Software Project Trade-off Model

Although CRF defines five components of risk not all of them need to be used for each perspective. For example, for tester perspective low *testing effectiveness* causes high *number of defects* and *defect rate*. When this occurs there are no mitigations available. In a real project a manager may decide to release a project later and perform additional testing. This would increase *testing effectiveness* and thus the product quality. However, SPTM is not a DBN so such time-series modeling is not available.

6. CONCLUSIONS

Traditional risk assessment usually does not support causal relationships quantitatively. Our proposed Causal Risk Framework, which in a basic form has two dimensions (components of risk and user perspectives) can be a useful extension for decision makers. Bayesian nets appear to be the modeling technique naturally capable of combining with CRF. Our analysis showed that many of the previously designed BNs in the software engineering area cannot be combined with CRF usually because of lack of necessary causal links.

However, we have restructured the previously developed Software Project Trade-off Model to be compatible with CRF by allowing three risk perspectives: analytic, programmer and tester. Such an application of CRF to SPTM results in a clearer and more understandable model structure; this is an important feature given the model size.

Our future work will focus on developing a software tool which will allow CRF to be applied to existing and new models. We also plan to analyze the applicability of CRF to predictive models in areas other than software engineering.

REFERENCES

[1]BAI C.G., HU Q.P., XIE M., Ng S.H., *Software failure prediction based on a Markov Bayesian network model*, Journal of Systems and Software, Vol. 74, No. 3, 2005, 275–282.

[2]BAYES T., *An essay towards solving a Problem in the Doctrine of Chances. By the late Rev. Mr. Bayes, F.R.S. communicated by Mr. Price, in a letter to John Canton, A.M.F.R.S.*, Philosophical Transactions of the Royal Society of London, Vol. 53, 1763, 370-418.

[3]BIBI S., STAMELOS I., *Software Process Modeling with Bayesian Belief Networks*, In: Proc. of 10th International Software Metrics Symposium, Chicago, 2004.

[4]COCKRAM T., *Gaining Confidence in Software Inspection Using a Bayesian Belief Model*, Software Quality Journal, Vol. 9, No. 1, 2001, 31–42.

[5]DABNEY J.B., BARBER G., OHI D., *Predicting Software Defect Function Point Ratios Using a Bayesian Belief Network*, Proceedings of 2nd International Workshop on Predictor Models in Software Engineering, Philadelphia, PA, 2006.

[6]FENTON N., HEARTY P., NEIL M., RADLIŃSKI Ł., *Software Project and Quality Modelling Using Bayesian Networks*, In: Artificial Intelligence Applications for Improved Software Engineering Development: New Prospects, F. Meziane, S. Vadera (eds.), IGI-Global, 2009 (in press).

[7]FENTON N., MARSH W., NEIL M., CATES P., FOREY S., TAILOR M., *Making Resource Decisions for Software Projects*, Proceedings of the 26th International Conference on Software Engineering, Washington, DC, IEEE Computer Society, 2004, 397–406.

[8]FENTON N., NEIL M., MARSH W., HEARTY P., RADLIŃSKI Ł., KRAUSE P., *On the effectiveness of early life cycle defect prediction with Bayesian Nets*, Empirical Software Engineering, Vol. 13, 2008, 499–537.

[9]FENTON N., NEIL M., *Measuring your Risks*, London: Agena, 2005.

[10]FENTON N., NEIL M., *Visualising your Risks*, London: Agena, 2005.

[11]FENTON N.E., NEIL M.., MARSH W., KRAUSE P., MISHRA R., *Predicting Software Defects in Varying Development Lifecycles using Bayesian Nets,* Information and Software Technology, Vol. 43, No. 1, 2007, 32–43

[12]HEARTY P., FENTON N., MARQUEZ D., NEIL M., *Predicting Project Velocity in XP using a Learning Dynamic Bayesian Network Model*, IEEE Transactions on Software Engineering, Vol. 37, No. 1, 2009, 124–137.

[13]JONES W., GALLO A., *A Process-Based Approach to Handling Risks,* IT Professional, Vol. 9, 2007, 10–15.

[14]MERRIAM-WEBSTER, *Merriam-Webster's Online Dictionary*, www.merriam-webster.com, 2009.

[15]PAI G., BECHTA-DUGAN J., LATEEF K., *Bayesian Networks applied to Software IV&V*, Proceedings of 29th Annual IEEE/NASA Software Engineering Workshop, IEEE Computer Society, Washington, DC, 2005, 293-304.

[16]PAI G.I., DUGAN J.B., *Empirical Analysis of Software Fault Content and Fault Proneness Using Bayesian Methods*, IEEE Transactions on Software Engineering, Vol. 33, No. 10, 2007, 675-686.

[17]PROJECT MANAGEMENT INSTIUTE, *A Guide to the Project Management Body of Knowledge. Third Edition,* American National Standard ANSI/PMI 99-001-2004, Project Management Institute, 2004.

[18]RADLIŃSKI Ł., FENTON, N., NEIL, M., MARQUEZ, D., *Improved Decision-Making for Software Managers Using Bayesian Networks*, In: Proc. of 11th IASTED International Conference Software Engineering and Applications, Cambridge, MA, 2007, 13–19.

[19]RADLINSKI L., *Improved Software Project Risk Assessment Using Bayesian Nets*, Ph.D. Thesis, Queen Mary, University of London, London, 2008.

[20]RADLIŃSKI Ł., *Integrating Data Sources in a Software Project Risk Assessment Model,* Research Papers of University of Economics in Wrocław. Knowledge Acquisition and Management, 2009 (submitted).

[21]SIMON P., HILLSON D., NEWLAND K. (eds.), *Project risk analysis and management (PRAM) guide*. The Association for Project Management, 1997.

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

[23]TEXAS DEPARTMENT OF INFORMATION RESOURCES, *Generic Software Project Risk Factors*, http://www.dir.state.tx.us/pubs/framework/gate2/projectplan/supplemental/, 2008.

[24]WAGNER S., *A Bayesian network approach to assess and predict software quality using activity-based quality models*, Proceedings of the 5th International Conference on Predictor Models in Software Engineering, New York, ACM Press, 2009.

[25]WARD S., CHAPMAN C., *Transforming project risk management into project uncertainty management*, International Journal of Project Management, Vol. 21 No. 2, 2003, 97–105.

[26]WOOFF D.A., GOLDSTEIN M., COOLEN F.P.A., *Bayesian Graphical Models for Software Testing*, IEEE Transactions on Software Engineering, Vol. 28, No. 5, 2002, 510–525.

[27]ZHOU Y., WÜRSCH M., GIGER E., GALL H.C., LÜ J., *A Bayesian Network Based Approach for Change Coupling Prediction*, Proceedings of the 15th Working Conference on Reverse Engineering, Washington, DC, IEEE Computer Society, 2008, 27–36.