The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 2022

Volume 17, Pages 120-128

ICRETS 2022: International Conference on Research in Engineering, Technology and Science

Implementation of Essence Practice into Bayesian Networks

Lidiya IVANOVA
Tomsk State University

Denis ZMEEV
Tomsk State University

Oleg ZMEEV
Tomsk State University

Abstract: Modern project management in software engineering still lacks a commonly accepted applicable formal model of projects that can be used for accumulating experience or for developing applied optimization methods that can potentially help IT companies to reduce risks and deliver software in a more efficient way. The Essence language which was developed by SEMAT initiative can potentially work as such a formal model for software projects, but current Essence practitioners mostly focus on methodology work for describing different approaches to perform tasks for software projects. In this work, we propose a way to develop a prototype of a decision support system based on the Essence kernel and language in combination with an applied optimization model. In order to do this, we firstly design how to include Essence practice in an applied math model and then modify a Bayesian network that finds false-positive manager mistakes. As an example of implementation of the achieved results, we present the current state of the plugin for the software project management system Redmine that uses our approach to help managers of projects.

Keywords: Bayesian network, Essence, Project Management, Redmine, SEMAT

Introduction

Statistics provided by the Standish Group in the new CHAOS Report (Portman 2020: Beyond infinity», 2021) show that only about 31% of projects succeed, 50% cause difficulties, and 19% fail, with this value barely changing in recent years. Statistics collected from sources (Project Smart, 2009; Wojewoda & Hastie, 2015) are shown in Table 1. From these data, it can be concluded that the IT industry is an area with a high proportion (about 70%) of outcomes that cannot be called completely successful. Given the rapid growth of the industry and its impact on the economy, the need to increase the proportion of successful projects, and hence the scope of works devoted to the study of this problem, is relevant.

Table 1. Project success statistics from CHAOS reports

	2011	2013	2015	2020
Successful	29%	31%	29%	31%
Challenged	49%	50%	52%	50%
Failed	22%	19%	19%	19%

Over the past 10 years, many tools have appeared in the field of software development that facilitate the technical component of the development process: new versions of integrated development environments, automatic code analysis and refactoring tools, new programming languages with concise syntax, various frameworks and libraries that can significantly increase the speed of software development and its quality.

- This is an Open Access article distributed under the terms of the Creative Commons Attribution-Noncommercial 4.0 Unported License, permitting all non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.
- Selection and peer-review under responsibility of the Organizing Committee of the Conference
© 2022 Published by ISRES Publishing: www.isres.org

However, the technological breakthrough apparently did not affect the percentage of successful completion of projects, and therefore it can be assumed that the main reason for failures is not in the technical component of the process. According to the author of Software Conflict 2.0: The Art and Science of Software Engineering (Glass, 2006), the reasons for failures in the computer industry are: inability to estimate (project deadlines, costs) due to a lack of understanding of the process, instability of requirements for the software being developed, as well as "whims and delusions" – the desire of managers and developers to find universal solutions to all emerging problems. In the field of project evaluation and planning, as well as in the field of risk management in software development, significant changes have not occurred over the past decade. The results of various scientific studies on these topics have not found their application in business, or have not been widely disseminated.

A few years ago, our team started research in the field of application of mathematical models for project management. The result of this research is a prototype of the AI assistant for software management, which uses Bayesian networks as a basic mathematical model (Ivanova et al., 2021; Zmeev, 2022). Alpha state game allows manager to evaluate current status of the project. However, our approach allows us to find internal conflicts between statements about the status of the project. The manager must indicate which checkpoints are completed in his opinion, and the tool calculates the probability of the truth of the hypothesis that these checkpoints are completed. The prototype is developed as a plugin for the Redmine project management system.

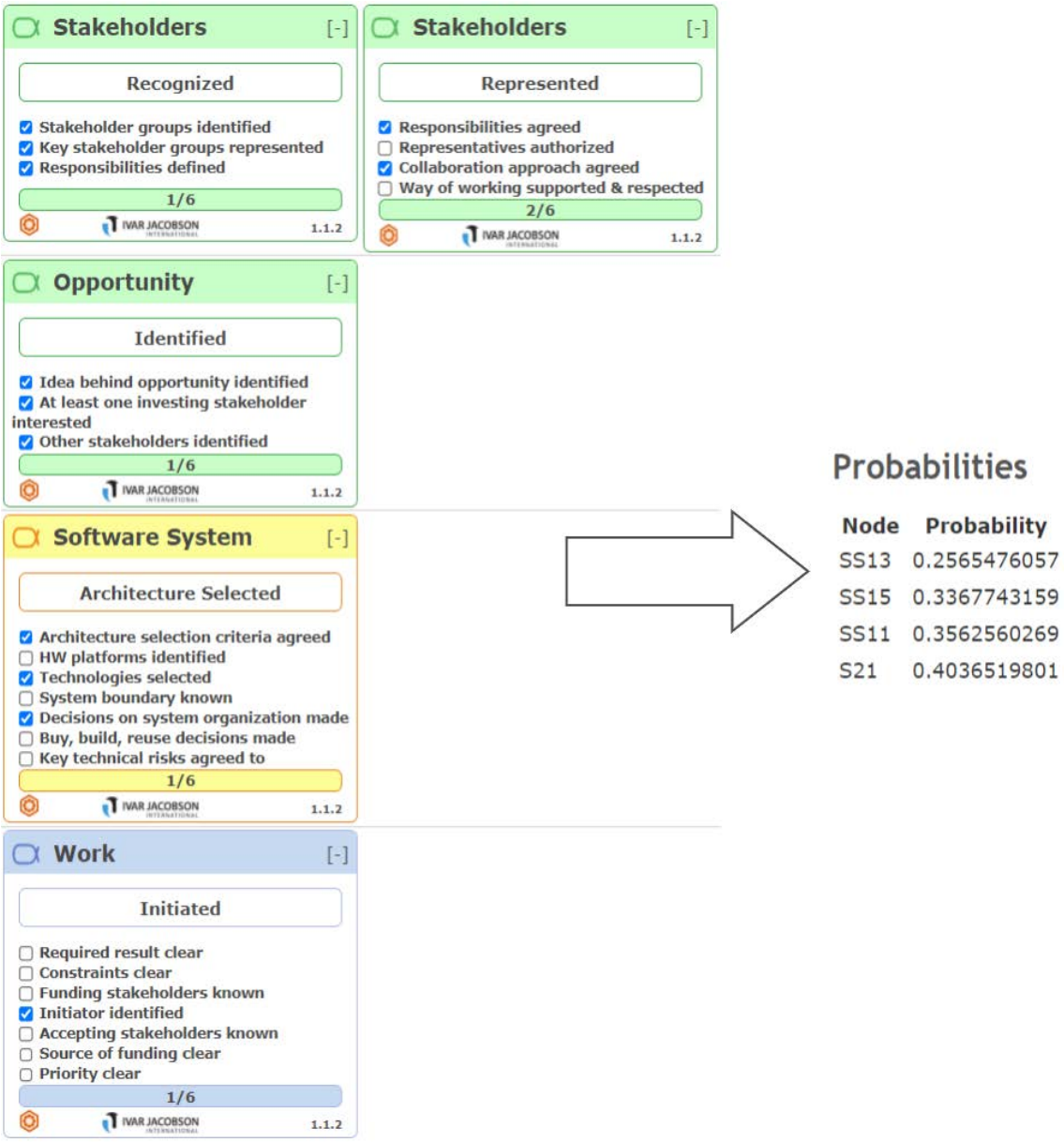

Figure 1. Prototype first version working results

Since the first version of the prototype uses only a subjective assessment – the manager's opinion in form of checkboxes from alpha state cards (Figure 1). In order to determine the probability of a manager's false-positive mistake more accurately, we decided to add accounting for objective evidence of the project progress – different work products that had been developed in the process of work. Examples of work products are source code, documentation, mockups, and so on. It should be noted that the impact of work products can be both positive and negative, since work products can be, for example, reports of errors, which has negative meaning. Adding new functionality will allow expanding the set of Essence language entities supported by the prototype.

This paper presents a method for accounting for work products in a Bayesian network and the probabilistic inference counting algorithm for determining the probability of the truth of a hypothesis corresponding to a statement about the state of a software development project. Section 2 presents a Bayesian network configuration designed to account for the presence of work products and their levels of details. Section 3 contains the algorithm developed for conditional probability counting, which takes into account both the positive and negative impact of the evidence, as well as the number of instances of work products corresponding to the evidence. Section 4 presents the results of calculations for a project that is at the initial stage of development.

Accounting Work Product in a Bayesian network

Work products, their number, and levels of detail change throughout the life cycle of a project, so it is necessary to dynamically rebuild the Bayesian network at each iteration. As an example, consider a project in which 4 work products «Antipattern Report» were created. This work product is evidence of an antipattern in the system – a common inefficient solution to a commonly occurring design and/or implementation problem. The levels of detail achieved are shown in Figure 2. Achieved levels of detail are marked with a checkmark.

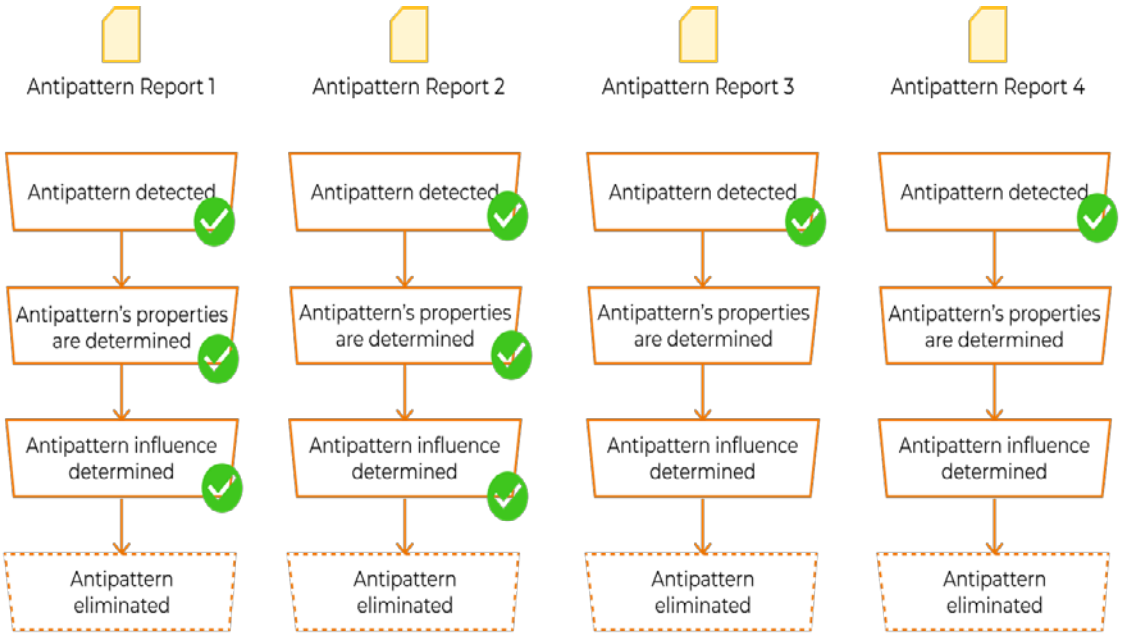

Figure 2. Work products and their levels of detail

The presence of such work products affects the statement about the Software System «Key technical risks are agreed», and the strength of evidence and its sign depends on the level of detail. In the first level of detail, Antipattern Report has a negative meaning because it is a potentially future technical risk. However, if antipattern influence determined Antipattern Report become positive evidence because we have more information about this part of the code.

In the process of research, various configurations of nodes and edges were considered to display information about work products. Figure 3 shows the resulting Bayesian network configuration for the above example. This configuration involves creating a node for each level of detail that affects a specific state node and the corresponding dependency.

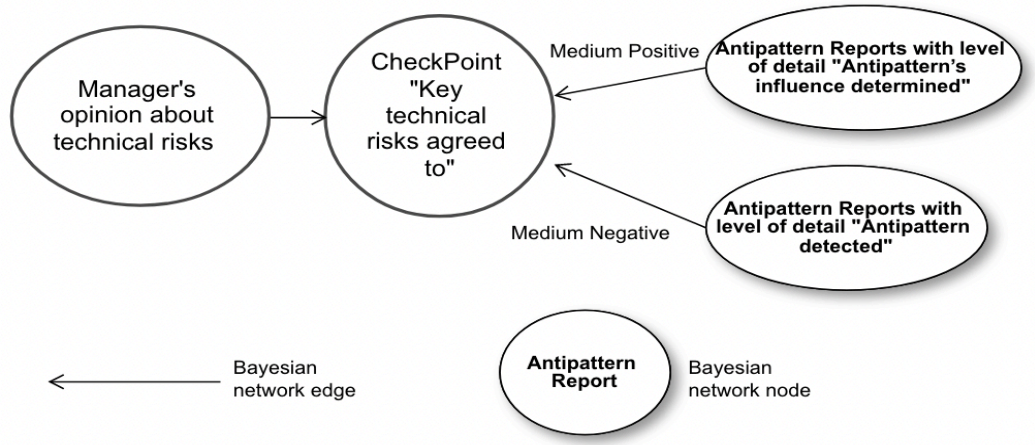

Figure 3. Bayesian network nodes and edges for Antipattern Reports accounting

This configuration has the following advantages:

- slow increase in the number of nodes when adding new work products and practices. Since nodes are being created for the levels of detail, the rate of increase in their number and the number of dependencies will be rather low;
- a small number of edges to the state nodes of the project. The experience of using the prototype has shown that if a node has more than 40 parent nodes, the calculation of a probabilistic inference takes several hours. In this regard, it is necessary to reduce the number of dependencies as much as possible. At the moment, there are nodes in the system that have 15 parent nodes.

The disadvantage of this configuration is the need to determine the strength of the evidence of the added nodes since it is necessary to take into account the number of work products that have reached the appropriate level of detail. Dynamic rebuilding of the Bayesian network creates a problem of obtaining and storing conditional probabilities. In the first version of the prototype, conditional probabilities were predetermined and stored as a probability vector. The fact is that with a large number of practices, the size of the vector increases very quickly, which negatively affects both performance and the memory used. In addition, the vector cannot be predetermined because the set of practices may change during project development. Because work products can have a negative impact on the project progress, handling for this impact needs to be added. Also, keep in mind that work products are not unique. For example, requirements analysis creates many use case diagrams. Therefore, it is necessary to take into account the number of work products of a certain type.

Conditional Probability Counting Algorithm Modification

The main difficulty is the determining conditional probabilities. When adding practices supported by the team, it is not possible to build a vector in advance, since the number of work products, and their levels of detail change with each iteration of the project. In addition, the team can combine several practices. Therefore, it was decided to abandon the storage of conditional probabilities and calculate them during the operation of the main algorithm.

Consider the case shown in Figure 2. It is necessary to determine the probability that the node corresponding to the Software System «Key technical risks agreed to» alpha checkpoint is true, that is, the probability that the hypothesis that all key technical risks of the project are agreed is true. Let us denote this hypothesis as H, and denote the opposite hypothesis that this checkpoint is false, denote it as ¬H. The presence of work products with a level of detail is evidence that can affect the likelihood of our hypothesis being true. Let us denote the evidence corresponding to the level of detail «Antipattern detected» as e_1, the evidence corresponding to the level of detail «Antipattern's influence determined» as e_2. There is an interpretation of the influence of evidence on the change in the ratio of conditional probabilities of mutually exclusive hypotheses H and ¬H, presented in the form of a line, the center corresponds to the ratio of equiprobability of these two hypotheses (Bayes, n.d.). The appearance of evidence shifts the pointer toward the hypothesis favored by the evidence by an amount corresponding to the strength of the evidence's influence.

Figure 4 shows an example from the specified source, in which the ratio of prior probabilities of hypotheses is 1, the circles indicate the occurrence of evidence with a 2:1 chance in favor of hypothesis H. It was decided to use this approach to determine the conditional probabilities for the Bayesian network nodes instead of storing the conditional probabilities as a probability vector.

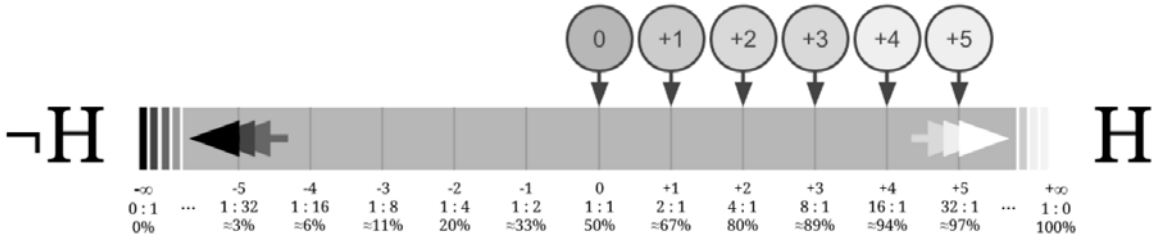

Figure 4. Log-odds line («Bayes' rule: Log-odds form», n.d.)

For the convenience of calculations, we represent the change in the ratio of conditional probabilities as a power of two, that is, 2^k, where k is the strength of the influence of the evidence. Thus, stronger evidence will change the ratio more than weaker evidence. If evidence has a positive impact on the project (evidence is a positive factor in the development of the project), there are two possible cases: the true value of the evidence in favor of H will increase the conditional probability of the hypothesis H. A false value for evidence will decrease the probability of hypothesis H.

Since some work products, such as an antipattern report, can have a negative impact on the project, it is necessary to consider the appropriate cases:

1. Evidence affects negatively. The evidence is the truth. The probability that hypothesis H is true decreases.
2. Evidence affects negatively. The evidence is false. The probability of hypotheses does not change, since the absence of fixed problems or risks in the project does not mean that there is progress, nor that there are any obstacles to development.

With the approach described above, we can determine the conditional probability for positive and negative evidence. However, as mentioned earlier, the chosen Bayesian network configuration gives rise to the problem of determining the strength of influence. The strength of influence, in this case, should depend on the number of work products that have reached these levels of detail. At the same time, it is necessary to remember such indicators as the size of the project (small, medium, large), as well as some norms for the number of work products adopted by the creators of practices.

To account for project size and recommendations for the number of work products it's necessary to define new coefficients:

- project scale – a coefficient reflecting a subjective assessment of the ratio of the project size in relation to a typical project for a set of practices used. This parameter was introduced to ensure further scaling of the mathematical model for various projects.
- norm – the number of instances of the work product, characteristic of a typical project, according to the author of a particular practice.

These concepts have already been encountered in the software engineering literature. For example, A. Jacobson argues that « A really large system might have seven use cases» (Cockburn, 2016). The authors of (Jacobson et al., 1999) speaking of use-case models, state that «at the end of the design phase, these models are less than 10% complete. Standing apart are use-case and analysis models, which … can include significantly more (up to 80% in some cases)». Thus, the researchers also point to the existence of some standards that determine the number of work products for projects, depending on the scope of the project. Over fulfillment of such a «norm» is considered redundant, since the creation of an additional work product instance in excess of the «norm» does not make a major contribution to the project. Let N be the number of work products of a particular type with the level of detail being considered. The following situations are possible:

1. N = 0. The strength of evidence does not change.
2. N < norm * scale. It is necessary to reduce the strength of evidence.

3. N = norm * scale. The strength of evidence does not change.
4. N ≥ norm * scale. The power of evidence is increasing. In the case of exceeding the product of the norm by the scale, the increase in the strength of evidence should be insignificant, since the excess of the number of work products of the recommended standards is more likely evidence of overengineering and formalism than the basis for a significant increase in the strength of evidence.

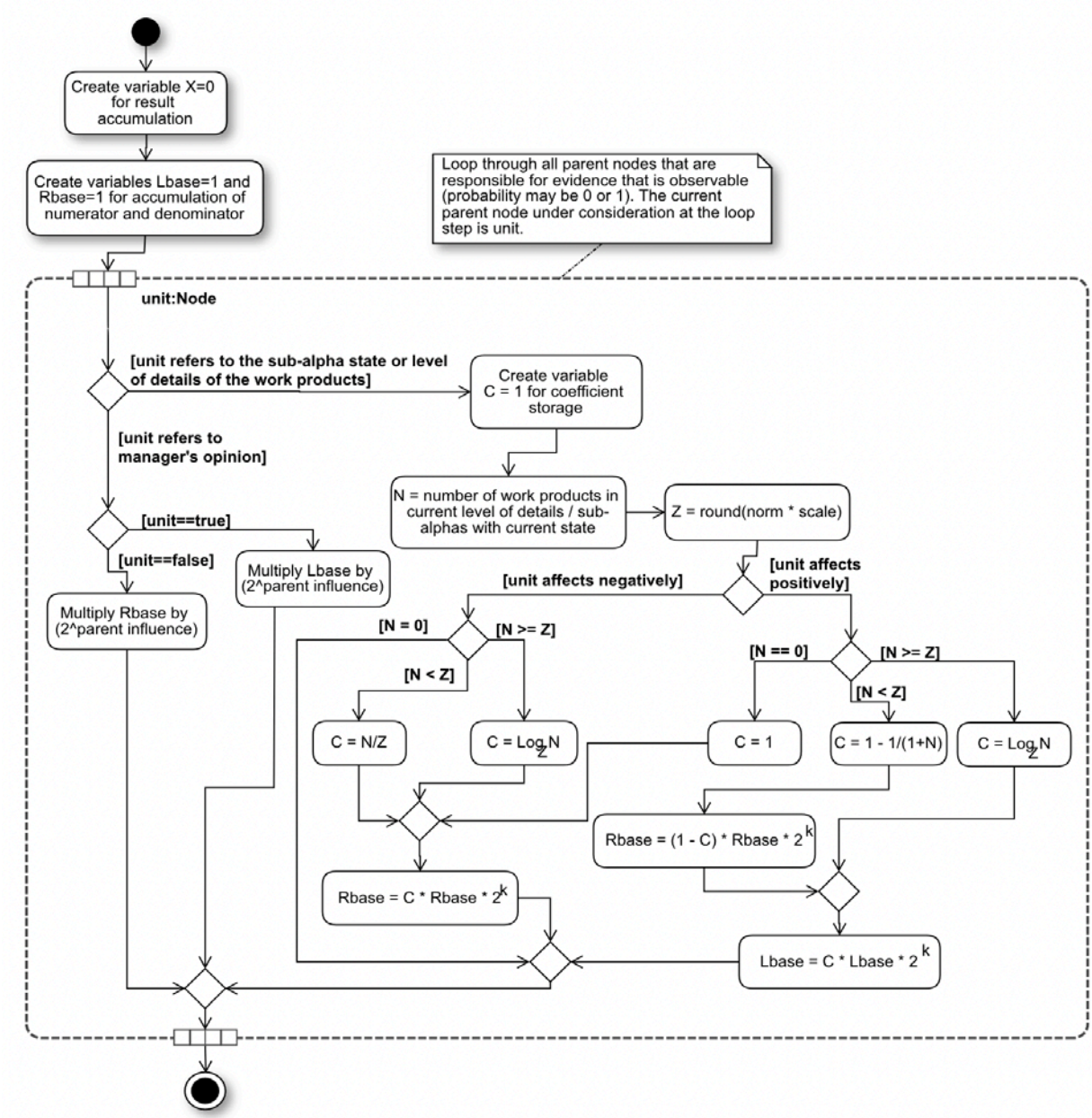

Figure 5. Parents corresponding to work products and sub-alphas

Let us designate the coefficient of change in the strength of influence as C. The value of C for evidences with positive impact is determined by the formula (1):

$$C = \begin{cases} 1, if\ N = 0 \\ 1 - \frac{1}{1+N}, if\ N < norm * scale \\ 1, if\ N = norm * scale \\ log_{norm*scale} N, if\ N > norm * scale \end{cases} \quad (1)$$

The value of C for evidences with negative impact is determined by the formula (2):

$$C = \begin{cases} 1, if\ N = 0 \\ \dfrac{N}{norm * scale}, if\ N < norm * scale \\ 1, if\ N = norm * scale \\ log_{norm*scale} N, if\ N > norm * scale \end{cases} \quad (2)$$

Since all aspects related to the addition of work product accounting have been determined, it is necessary to consider modifying the algorithm for calculating conditional probability. The algorithm can be divided into two steps. In the first stage (Figure 5), all parent nodes that correspond to work products and sub-alphas are processed. The second one (Figure 6) considers the parent nodes corresponding to the alphas of the project.

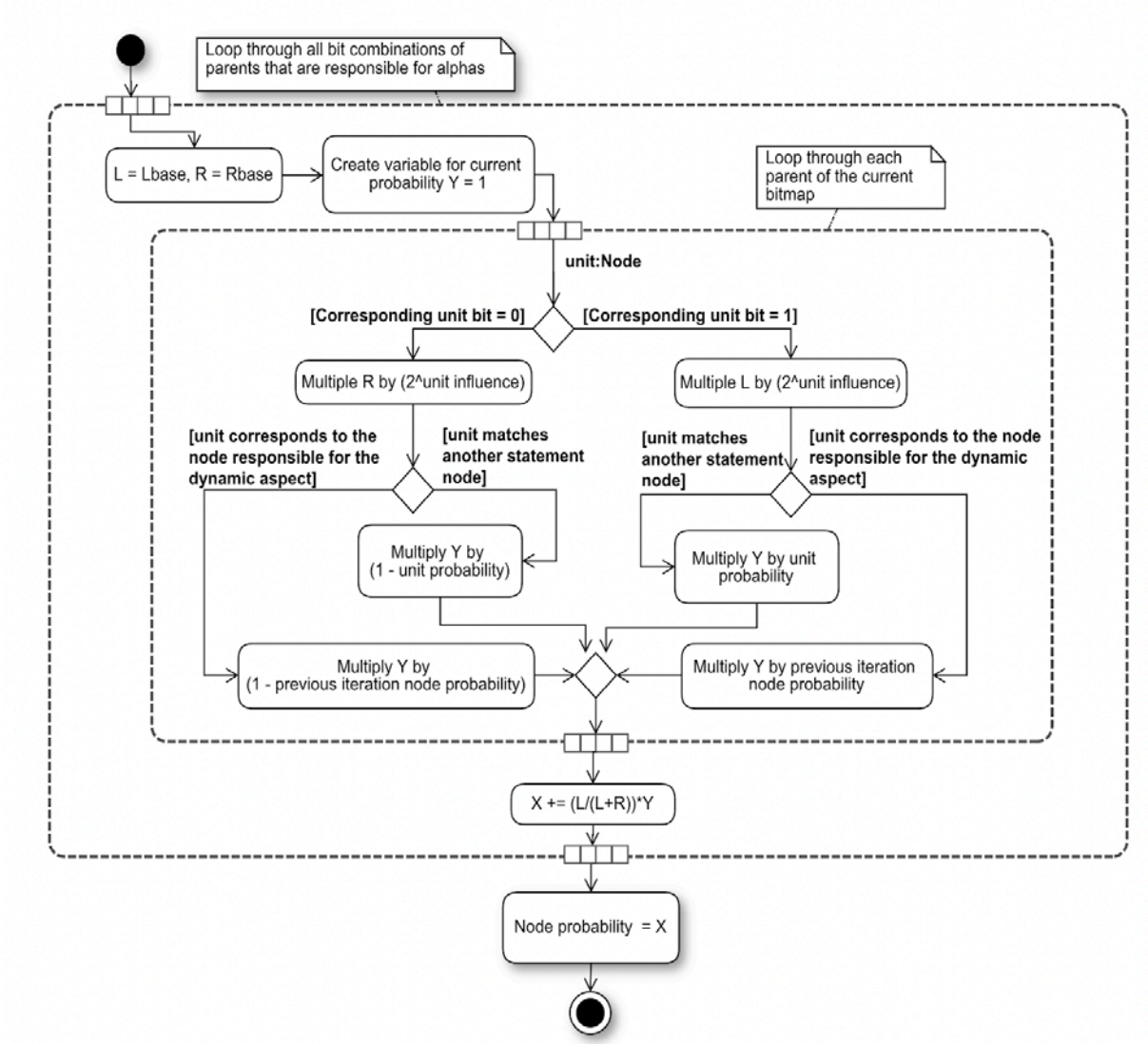

Figure 6. Parents corresponding to alphas

Prototype Working Results

Figure 7 shows the results of calculations made using the new version of the evolutionary prototype for one iteration in the absence of data on antipatterns. At the same time, the manager believes that the «Key technical risks agreed to» checkpoint is set to «true». The value of the checkpoint «Decision on system organization

made» is highlighted in red, because, according to the manager, it has the value «true», however, according to calculations, the probability of the system being in this state is less than the threshold value of 0.5.

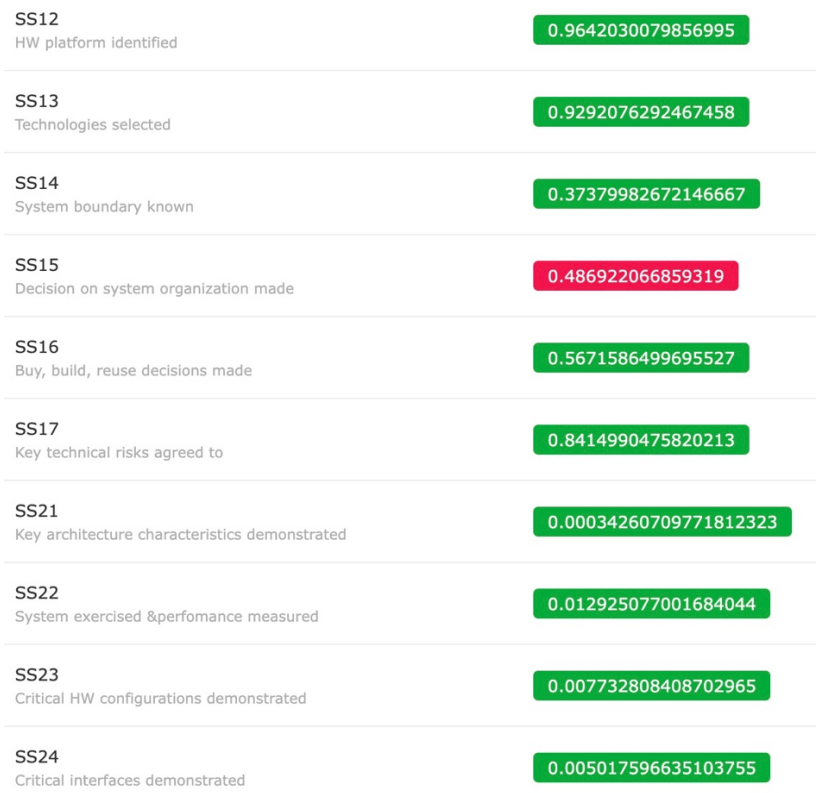

Figure 7. Results of calculations in the absence of data on antipatterns

Table 2 shows the values of the probability of truth for the Software System «Key technical risks agreed to» checkpoint, which can be affected by the presence of antipatterns in the system. The test project is at the initial stage of development.

Table 2. Probability values for checkpoint «Key technical risks agreed to»

Case	«Key technical risks agreed to» probability
Information about work products is not counted	0,841499
4 work products with a negative impact	0,662517
2 work products with a negative impact, 2 with positive impact	0,88555
4 work products with a positive impact	0,90467
16 work products with a negative impact	0,629207

The results of the calculations demonstrate the impact that the presence of work products has on the probability of the project being in a certain state. This information adds an objective dimension that helps improve the accuracy of the estimate given by the prototype.

Conclusion

The second version of the system allows us to take into account objective evidence of project progress. Thus, the manager does not need to constantly check which work products have appeared in the system and what their status is. The applied method for determining conditional probabilities made it possible to abandon the storage of probability vectors, which had a positive effect on the memory used.

We see the following directions of development for the proposed approach: the dependence functions of the strength of the influence of evidence on the number of evidence can have a different form, and it is necessary to take into account the degree of the formalism of the method used. For example, the Unified Process is a more

rigorous method than Scrum. In addition, the development of the tool requires the creation of a repository of practices, as well as additional research on the issue of Bayesian network learning based on statistical data.

Scientific Ethics Declaration

The authors declare that the scientific ethical and legal responsibility of this article published in EPSTEM journal belongs to the authors.

Acknowledgements or Notes

This article was presented as an oral presentation at the International Conference on Research in Engineering, Technology and Science (www.icrets.net) conference held in Baku/Azerbaijan on July 01-04, 2022.

References

Bayes (n.d.) *Bayes' rule: Log-odds form.* Arbital. Retrieved March 1, 2022, from https://arbital.com/p/bayes_log_odds/

Cockburn, A. (2016). *Writing effective use cases*. Addison-Wesley.

Glass, R. L. (2006). *Software conflict 2.0: The art and science of software engineering*. Developer.

Ivanova, L. S., & Rafikova, R. R., & Zmeev, D. O. (Eds.). (2021). *Presenting the progress of a software development project in the form of a dynamic Bayesian network*. Information technologies and mathematical modelling (ITMM-2020). Proc. of the ninth international conference named after A.F. Terpugov, 291–297.

Jacobson, I., Booch, G., Aguilar, J. L., Pimentel, E., Rumbaugh, J., & Sánchez Salvador. (1999). *The Unified Software Development process*. Addison-Wesley.

Portman, H. (2021). *Review Standish Group – Chaos 2020: Beyond infinity*. Retrieved July 1, 2022, from https://hennyportman.wordpress.com/2021/01/06/review-standish-group-chaos-2020-beyond-infinity/

Project Smart (2009). *The curious case of the chaos report.* Retrieved March 1, 2022, from https://www.projectsmart.co.uk/it-project-management/the-curious-case-of-the-chaos-report-2009.php

Wojewoda, S., & Hastie, S. (2015, October 4). *Standish Group 2015 chaos report - Q&A with Jennifer Lynch*. InfoQ. Retrieved February 1, 2022, from https://www.infoq.com/articles/standish-chaos-2015/

Zmeev, D. O. (2022). *Decision support system prototype for project management based on OMG Essence Standard and Bayesian networks.* [Candidate's dissertation]. Tomsk State University.

Author Information

Lidiya Ivanova
Tomsk State University
36, Lenin Avenue, Tomsk, 634050, Russia
Phone.: +7 (382-2) 52-97-93
Contact e-mail: *lidiya.ivanova@persona.tsu.ru*

Denis Zmeev
Tomsk State University
36, Lenin Avenue, Tomsk, 634050, Russia
Phone.: +7 (382-2) 52-97-93
Contact e-mail: *denis.zmeev@accounts.tsu.ru*

Oleg Zmeev
Tomsk State University
36, Lenin Avenue, Tomsk, 634050, Russia
Phone.: +7 (382-2) 52-97-93
Contact e-mail: *ozmeyev@gmail.com*

To cite this article:
Ivanova, L.S., Zmeev, D.O. & Zmeev, O.A. (2022). Implementation of essence practice into Bayesian networks. *The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 17,* 120-128.