A Bayesian Network for Evaluating Software Project Team Fit: An Industrial Case Study

Felipe Cunha
Federal University of Campina Grande (UFCG)
Campina Grande, Brazil
felipe.cunha@virtus.ufcg.edu.br

Mirko Perkusich
Federal University of Campina Grande (UFCG)
Campina Grande, Brazil
mirko@virtus.ufcg.edu.br

Danyllo Albuquerque
Federal University of Campina Grande (UFCG)
Campina Grande, Brazil
danyllo.albuquerque@virtus.ufcg.edu.br

Kyller Gorgônio
Federal University of Campina Grande (UFCG)
Campina Grande, Brazil
kyller@virtus.ufcg.edu.br

Angelo Perkusich
Federal University of Campina Grande (UFCG)
Campina Grande, Brazil
perkusic@virtus.ufcg.edu.br

ABSTRACT

[Context] Forming effective software development teams requires balancing both technical and collaborative factors. While optimization methods—such as genetic algorithms—have been applied to automate team formation, they typically rely on fitness functions built from fixed weights or handcrafted rules. These ad hoc formulations rarely reflect structured expert reasoning and may misalign with organizational expectations. **[Objective]** This study focuses on constructing a Bayesian Network (BN) that formally encodes expert knowledge about team-project fit. The BN is intended to serve as a reusable, interpretable fitness function in future optimization-based team formation systems. **[Method]** We conducted a case study in a large software organization with over 300 employees. Through expert workshops, we identified relevant developer capabilities, defined qualitative states, and structured the BN accordingly. Conditional Probability Tables (CPTs) were populated using expert-labeled what-if scenarios, then calibrated using the Ranked Nodes method. **[Results]** The final BN comprises nodes for technological and collaboration fit. Validation against held-out, expert-labeled scenarios yielded consistently low Brier Scores, demonstrating strong alignment with expert expectations. **[Conclusion]** BNs offer a viable, interpretable mechanism for encoding expert knowledge in team formation. This work establishes a structured and transparent foundation for future integration into automated team formation workflows.

KEYWORDS

Bayesian Networks, Software Team Formation, Software Engineering, Knowledge Engineering, Expert Systems.

1 Introduction

Forming effective software project teams involves balancing technical needs, collaboration dynamics, and organizational constraints. While automated team formation methods—often based on genetic algorithms—have improved search efficiency [1], they typically rely on ad hoc fitness functions with fixed weights or hand-tuned heuristics. Such functions rarely reflect expert judgment, leading to opaque, brittle evaluations that may misalign with organizational goals.

This limitation reflects a broader challenge in artificial intelligence (AI): purely data-driven methods often struggle in low-data or high-stakes contexts [17]. In response, knowledge-guided paradigms, including physics-informed neural networks (PINNs) [14], embed expert knowledge into model design. While our approach does not involve statistical learning, it adopts this spirit by structuring a Bayesian Network (BN) based on expert reasoning. This BN is intended to serve as a fitness function within an optimization-based team formation framework, such as a genetic algorithm.

In this paper, we focus on the design and evaluation of a BN that captures expert reasoning about team-project fit. While the BN is intended to serve as a fitness function within a broader team formation system (e.g., based on a genetic algorithm), this paper centers exclusively on the elicitation, structuring, and validation of the BN itself. The model encodes both technical and collaborative factors in a transparent and interpretable manner.

We report a case study in a large software organization (>300 employees) where one expert collaboratively defined and calibrated the BN. Evaluation against held-out, expert-labeled scenarios yielded low Brier Scores, demonstrating strong agreement with expert expectations.

The rest of this paper is structured as follows: Section 2 reviews related work; Sections 3 and 4 describe methods and findings; Section 5 discusses limitations, and Section 6 concludes with directions for future work.

2 Background and Related Work

This section situates our approach within key research areas: expert-engineered BNs and software team formation.

2.1 Bayesian Networks in Software Engineering

BNs are probabilistic graphical models that represent conditional dependencies among variables via a directed acyclic graph. Each node corresponds to a variable, and edges encode causal or influential relationships. BNs support reasoning under uncertainty and are widely used in software engineering for tasks such as defect prediction, effort estimation, and strategic indicator modeling [11]—especially when data is scarce.

The Knowledge Engineering of Bayesian Networks (KEBN) process [13] provides a process for constructing BNs from expert input and automatic algorithms. It comprises three stages: (i) structural development, (ii) parameter estimation, and (iii) model validation.

A central component of any BN is the Conditional Probability Table (CPT), which defines the probability distribution of a child node given combinations of parent states. When variables are ordinal (e.g., “Low,” “Medium,” “High”), CPTs can be constructed using the ranked nodes method [5]. This method constrains CPTs to

follow monotonicity assumptions and supports four aggregation strategies: *weighted mean (WMEAN)*, *weighted minimum (WMIN)*, *weighted maximum (WMAX)*, and a hybrid form, *MIXMINMAX*. These capture different beliefs about how input factors interact (e.g., whether a single weak skill should dominate or if redundancy adds value). Our model adopts the KEBN framework and calibrates CPTs via the ranked nodes method.

2.2 Software Team Formation

Automatic team formation in software engineering has been widely studied, especially through genetic algorithms (e.g., [1–3]). However, existing genetic algorithms are based on fitness functions built ad hoc or purely empiric and do not formally incorporate expert reasoning. As a result, they risk being under-informed, constraining search effectiveness and adaptability.

More recent methods apply variational Bayesian neural networks to learn latent team representations from collaboration graphs and task skill requirements [9]. While these methods demonstrate quantitative improvements, they are computationally heavy and offer limited transparency.

In contrast, our approach embeds a BN constructed from expert knowledge as the fitness function within a genetic algorithm pipeline. This design combines the strengths of expert-informed reasoning and optimization efficiency, addressing both interpretability and knowledge integration, aligning with current trends in hybrid AI [17].

3 Methodology

This study investigates whether a BN, constructed from expert knowledge, can effectively model team-project fit in software engineering. The broader goal is to assess its viability as an interpretable fitness function within team formation algorithms. The central hypothesis is that a BN, if aligned with expert reasoning about team composition and project demands, can serve as a transparent and practical heuristic for guiding team selection.

We evaluate this hypothesis through the question: *Does the BN generalize well to unseen team configurations, according to the expert's expectations?* The focus is not on live performance optimization, but on whether the model meaningfully captures expert judgments across diverse hypothetical scenarios.

We conducted a KEBN-guided evaluative case study in a large, mature software organization, following three phases: (1) structure elicitation via expert workshops; (2) parameter estimation using synthetic training scenarios; and (3) validation with separate expert-labeled scenarios.

This case-based approach supports in-depth, contextual exploration of how expert knowledge can be formalized into a probabilistic model. Section 3.1 outlines the organizational setting, and Section 3.2 details the BN construction and validation procedures.

3.1 Case Context and Modeling Scope

The case study was conducted at a university-affiliated research and innovation center located in Brazil, specializing in information and communication technologies. The center employs over 300 engineers and researchers and executes more than 30 concurrent projects annually. Its portfolio includes short- to medium-term R&D&I initiatives in embedded systems, artificial intelligence, web and mobile development, and information security. These projects are funded either through public research grants or through direct industry contracts, with some initiatives combining both sources. They serve firms ranging from startups to large enterprises.

Functioning under a hierarchical, project-based organizational model, the center maintains a dedicated quality department responsible for process definition, auditing, and measurement. Although it does not hold formal certifications such as CMMI or ISO 9001, the center exhibits high process maturity. Agile methods, particularly Scrum and Kanban, are widely adopted and supported by an in-house platform for managing requirements, testing, issue tracking, and traceability.

The domain expert central to this study is the center's Chief Technology Officer (CTO), who has over 25 years of experience in the software development sector, including roles in multinational corporations, startups, and R&D&I environments. He has been closely involved with the initiative from its inception, providing domain framing aligned with industry needs, facilitating access to organizational data, and actively participating in every phase of the Bayesian Network development. His role in championing the solution reflects the design science paradigm, wherein collaborative problem-solving with practitioners is essential to generating relevant and applicable artifacts.

The modeling scope was intentionally limited to variables that characterize a developer's capability to contribute effectively to a software project. These include dimensions that are observable, experience-based, and relevant across diverse project contexts. Broader logistical and organizational factors—such as personnel availability, budget constraints, client preferences, and scheduling—were explicitly excluded. According to the expert, such elements are highly situational and are more appropriately considered during the final decision-making stage, after candidate teams have been recommended.

It is also important to distinguish between the BN itself and the mechanisms used to populate its input variables in operational contexts. For example, assessing collaborative fit requires access to historical performance data to infer pairwise compatibility. While such data collection and preprocessing steps are essential for applying the model in practice, they are beyond the scope of this paper. Our focus is limited to the structure of the BN and its evaluation using expert-labeled scenarios.

3.2 Bayesian Network Construction

This section describes how the Bayesian Network was constructed based on expert knowledge, guided by KEBN. The modeling process was organized into three main phases: structure development, parameter estimation, and model validation. Section 3.2.1 details how the network structure was defined through a series of expert workshops; Section 3.2.2 describes the procedures used to estimate the CPTs based on scenario data; and Section 3.2.3 explains how the resulting model was evaluated in terms of predictive accuracy and alignment with expert judgment.

3.2.1 Structure Development

The overarching objective of the BN was to estimate the *Team Fit* between a candidate software team and a given project. To guide the modeling effort, we adopted the Goal–Question–Metric (GQM) approach, which supports the systematic decomposition of abstract goals into measurable indicators. Given the modeling scope defined in Section 3.1, we applied GQM to translate the concept of team fit into operational aspects of developer capabilities that were considered both observable and contextually relevant. This

reasoning framework provided the conceptual backbone of the BN, identifying the main variables and intermediate constructs to be included.

We then refined this initial structure using established BN modeling idioms, d-separation considerations, and by evaluating data availability. This analysis informed the directionality of causal links, which was determined to support plausible reasoning paths and scenario-based inference. Each arc was justified through expert feedback and the logical relationships among variables.

With the network topology defined, we proceeded to specify the values/states for each variable. For each node, we documented the semantic interpretation and clarified the intended meaning of each possible value/state. These definitions were informed by expert insights and aligned with the terminology and evaluation practices commonly used within the organization.

3.2.2 Parameter Estimation

The second phase of the modeling process involved populating the CPTs for each node in the BN. For each target variable, we constructed a series of structured "what-if" scenarios, where the target's parent nodes were assigned specific value combinations. For example, consider a variable X with two parent nodes, Y and Z, each taking boolean values. A scenario in this context would represent the conditional distribution $P(X = \text{true} \mid Y = \text{true}, Z = \text{false})$ and $P(X = \text{false} \mid Y = \text{true}, Z = \text{false})$. These scenarios were designed to elicit the expert's judgment about the probability of each outcome under clearly defined input conditions.

The strategy for defining the set of scenarios (training dataset) varied depending on the characteristics of each node (e.g., number of parents, nature of the variable). These design choices are further discussed in Section 4.2 when presenting the elicitation outcomes.

Rather than collecting numerical probabilities directly, we adopted the verbal-numerical probability scale proposed by Renooij et al. [15] to reduce bias and improve interpretability. This scale maps qualitative expressions (e.g., "unlikely," "expected," "certain") to approximate numeric values and was presented as a reference during elicitation (see Table 1).

Table 1: Verbal-numerical probability scale (adapted from Renooij et al. [15]).

Verbal Expression	Numerical Equivalent (%)
Impossible	0
Improbable	15
Uncertain	25
Fifty-fifty	50
Expected	75
Probable	85
Certain	100

To illustrate the procedure, consider a simplified example in which the child node AE (Team Fit) has two parent nodes—AT (Technological Fit) and AC (Collaboration Fit)—each defined on a five-point Likert scale ranging from Very Low (VL) to Very High (VH). For a selected set of scenarios—i.e., unique combinations of AT and AC values—the expert was asked to assess the probability distribution over the five possible AE outcomes using the verbal scale.

Because the verbal-to-numerical mappings were provided independently for each outcome, the resulting probability estimates for a given scenario did not necessarily sum to 1. To ensure that each row in the CPT represented a valid probability distribution, we applied a normalization step, rescaling the values proportionally so that their total equaled 1.

To support expert reflection and validation, we visualized the resulting distributions using three complementary techniques:

- **Heatmap**: A matrix-based table color-coded the estimated probabilities across scenarios (see an example in Table 2), with darker green indicating higher probability and darker red indicating lower. This view enabled fast comparison across scenario configurations.
- **Bar charts**: For each scenario, we plotted the normalized probability distribution over the target variable's possible values. This allowed the expert to visually assess whether the resulting shapes aligned with domain expectations. Figure 1 shows an example for AE.
- **Mode extraction**: We also computed the modal outcome for each scenario to highlight the most likely value, supporting a rapid, high-level review of the CPT behavior.

Table 2: Example heatmap for the elicited probabilities for AE given its parents, AT and AC. Green indicates higher probabilities.

Parents		AE				
AT	AC	VL	L	M	H	VH
VL	VH	85	100	85	25	15
VH	VL	50	75	100	50	15
VL	VL	100	100	85	15	0
VH	VH	0	15	75	85	100
VL	M	50	75	85	25	15
M	VL	50	50	25	15	0
VH	M	25	50	75	85	100
M	VH	25	75	75	85	85

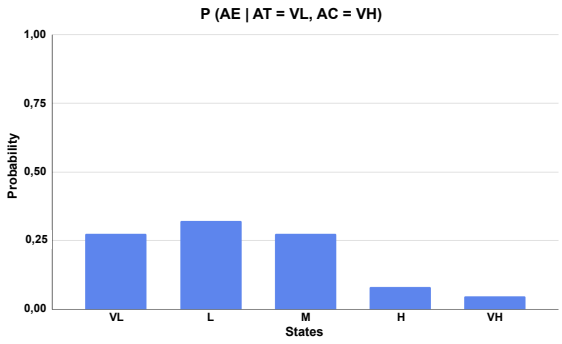

Figure 1: Normalized probability distributions over AE for selected parent configurations.

To derive the final CPTs, we used the expert-elicited scenarios as input to a specialized genetic algorithm designed to calibrate ranked node functions. This algorithm searches for the optimal function type (i.e., WMEAN, WMIN, WMAX, or MIXMINMAX), weights, and variance parameters that best fit the provided data, with the objective of minimizing the average Brier Score, a widely used measure for evaluating BN [4], across all "what-if" scenarios. The implementation is available in an open-source repository[1].

[1]https://anonymous.4open.science/r/RNMPython-3ABC/

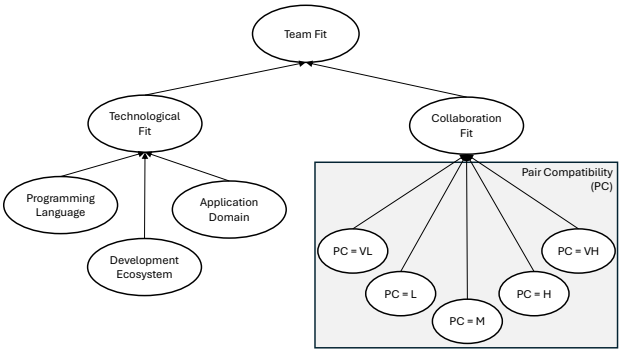

Figure 2: Final Bayesian Network structure for estimating team-project fit.

3.2.3 Model Validation

The modeling process was inherently iterative, involving multiple rounds of expert interaction as recommended by KEBN. Throughout the structure development phase, we conducted a series of workshops with the domain expert to progressively refine the network. After each major revision, we performed an expert walkthrough to review the structure, verify the directionality of causal links, and confirm the semantic validity of node definitions and state interpretations. This qualitative validation ensured that the model structure accurately reflected the expert's reasoning process and domain understanding.

To evaluate the model's predictive accuracy, we conducted a quantitative validation using a separate set of expert-labeled scenarios not employed during parameter estimation. Each scenario described a realistic team configuration and project context, based on situations previously experienced by the expert, and included an expert-assigned assessment of expected team fit. For each case, the BN generated a probability distribution over the possible outcomes for the target node (*Team Fit*). Predictive accuracy was measured using the Brier Score. Lower Brier Scores indicate stronger generalization to unseen configurations and better alignment with expert expectations.

4 Results and Discussion

This section presents the key outcomes of the modeling process, organized in alignment with the three sub-processes of KEBN: structure development, parameter estimation, and validation. First, we describe the final structure of the BN, highlighting how it operationalizes the notion of team-project fit based on expert reasoning. Next, we examine the CPTs, discussing how expert knowledge was encoded through scenario-based elicitation and supported by visualization techniques. Finally, we report the results of a predictive evaluation using a validation set of expert-labeled scenarios, quantifying the model's performance using the Brier Score and reflecting on its strengths and practical relevance.

4.1 Bayesian Network Structure

The BN developed in this study was designed to estimate the overall *Team Fit* between a proposed software team and a project. As explained in Section 3.1, the model scope was intentionally constrained to developer capabilities, excluding logistical or circumstantial factors. Within this scope, expert workshops helped identify the most relevant capability dimensions to include in the BN. Figure 2 shows the final network structure.

Given that the overarching modeling objective was to estimate *Team Fit* for a specific team-project configuration, this concept was designated as the target node in the BN. To decompose this abstract goal into measurable components, we collaborated with the expert to define intermediate questions that reflect key capability areas relevant to team success. We focused on two main categories of developer fit: *Technological Fit* and *Collaboration Fit*.

Technological Fit aggregates three dimensions identified by the expert as central to assessing technical alignment with a project: (i) familiarity with relevant programming languages, (ii) proficiency with the target development ecosystem (e.g., tools, platforms, APIs), and (iii) experience with the application domain (e.g., AI, mobile, embedded systems). Other technical aspects—such as DevOps maturity or testing practices—were intentionally excluded. According to the expert, these skills are typically distributed across members when forming cross-functional teams. Moreover, in this specific setting, such specialized capabilities often correlate with developer seniority (e.g., junior, senior), which is already accounted for in the system's genetic algorithm. Therefore, explicitly modeling these dimensions in the BN was considered redundant. The resulting model includes three parent nodes for *Technological Fit*: *Programming Language*, *Development Ecosystem*, and *Application Domain*.

Collaboration Fit, by contrast, captures dimensions of teamwork quality [6, 10], emphasizing behavioral and interpersonal dynamics. Instead of modeling this construct based on individual-level soft skills or personality traits—a common approach in team formation literature—we adopted a relational strategy grounded in historical team performance. The expert noted that collecting reliable soft skill profiles is not only expensive but also highly subjective, making them ill-suited for scalable modeling. In addition, empirical studies have questioned the predictive validity of soft skill and personality-based models in explaining team outcomes [8, 16]. This further motivated the use of performance-derived relational data as a proxy for collaborative aptitude.

In contrast, performance records from past projects are routinely tracked by mature organizations and offer a pragmatic data source. Based on this availability, we constructed a pairwise compatibility graph in which each node represents an individual, and edge weights reflect historical collaboration outcomes—such as project productivity or indicators of teamwork quality [6, 7, 10, 12]. These weights are then aggregated and used as inputs to estimate the likely collaboration effectiveness of candidate teams. Discussing details regarding the collaboration graph is out of the scope of this paper, since it extrapolates the BN's concerns. As shown in Figure 2, *Collaboration Fit* is influenced by the distribution of pairwise compatibility among team members. To model this without constraining the BN to a fixed team size, we adopted a fragment structure that captures the proportion of team member pairs classified in each compatibility level. This modeling strategy is detailed further in Section 4.2.

To ensure expressiveness and consistency across variables, we adopted a uniform five-point Likert scale ranging from *Very Low (VL)* to *Very High (VH)* for all nodes in the network. This decision was made in agreement with the domain expert to enable more nuanced distinctions between team characteristics, thereby increasing the precision of expert assessments during the elicitation phase. The use of a standardized ordinal scale also facilitated interpretation, comparison, and visualization of model outputs.

Given the ordinal nature of the variables, we employed the ranked nodes method. This technique provides flexibility for downstream integration: although a five-point scale was used during model construction, the distributions can later be collapsed into coarser granularity (e.g., three-point scales) without requiring re-elicitation—an important consideration for computational efficiency when embedding the BN into the genetic algorithm.

Each variable and its states were defined through collaborative workshops, where the expert provided detailed semantic descriptions for each level of the scale. This ensured shared understanding during the elicitation process and reinforced alignment with organizational practices. For instance, for the node *Technological Fit*, the state *Very Low (VL)* corresponds to a scenario where the team lacks any meaningful experience with the technologies required by the project, resulting in heavy reliance on external support and a high risk of delays and quality issues. A complete list of variable definitions and state descriptions is available in the supplementary material[2].

Although integration details are out of scope, we note that the *Programming Language*, *Development Ecosystem*, and *Application Domain* nodes were computed by comparing project needs with team members' profiles. A score was derived based on coverage of required technologies, with bonuses for redundancy across members.

4.2 Conditional Probability Tables

The CPTs were derived through expert elicitation. Table 3 presents the elicited data for the *Technological Fit* node. The table shows the final normalized probabilities across the five ordinal states (Very Low to Very High), reflecting the expert's expectations for different configurations of technical alignment.

Table 3: Elicited normalized probability values for *Technological Fit.*

			Technological Fit				
AD	**DE**	**PL**	**VL**	**L**	**M**	**H**	**VH**
VH	VH	VL	0.0833	0.1389	0.2778	0.4167	0.0833
VH	VL	VH	0.0000	0.0750	0.1250	0.3750	0.4250
VH	VL	VL	0.2239	0.2537	0.2985	0.1493	0.0746
VL	VH	VH	0.1515	0.0758	0.2576	0.2576	0.2576
VL	VH	VL	0.2128	0.3191	0.3617	0.1064	0.0000
VL	VL	VH	0.1818	0.1818	0.2727	0.3091	0.0545

The full elicitation data—including verbal-scale responses, intermediate heatmaps, and per-scenario bar charts—are provided in the supplementary material. The same procedure was followed for all other nodes in the network, with one exception: *Collaboration Fit.*

As discussed in Section4.1, because *Collaboration Fit* depends on the compatibility among all team members, it could not be modeled using a single *Pairwise Compatibility (PC)* node. Instead, we introduced a structural fragment composed of five nodes—`PC_VL`, `PC_L`, `PC_M`, `PC_H`, and `PC_VH`—each representing the proportion of team member pairs classified into a corresponding compatibility tier. For example, `PC_VL` represents the proportion of pairs with very low compatibility. To support integration into the BN, each `PC_*` node was discretized using the bins shown in Table 4.

[2]https://anonymous.4open.science/r/rnm-bn-validation-EC07/

Table 4: Discretization bins for `PC_*` nodes (applied to proportion values in [0, 1]).

Proportion range	Ordinal state
0–20%	VL
20–40%	L
40–60%	M
60–80%	H
80–100%	VH

As an example, consider a team composed of three members—A, B, and C—with PC(A, B) = 0.7, PC(A, C) = 0.6, and PC(B, C) = 0.5. Applying the binning rules:

- 0.7 and 0.6 are classified as H
- 0.5 is classified as M

This results in:

- `PC_H` = 2/3 → discretized as H
- `PC_M` = 1/3 → discretized as L
- All others = 0 → discretized as VL

This discretized profile serves as the input scenario for estimating the CPT of *Collaboration Fit.* Elicitation was conducted using representative profiles (e.g., "75% VH, 25% M") to gather expert judgments. As with other nodes, the resulting data was used to calibrate a ranked node function via a specialized genetic algorithm.

Table 5 shows an excerpt of the expert-elicited outcomes for *Collaboration Fit.* Unlike other nodes, its parents describe the distribution of pairwise compatibility, not individual characteristics, enabling the model to generalize across teams of varying sizes.

Table 5: Excerpt from elicitation table for *Collaboration Fit.*

			Collaboration Fit				
% VH	**% M**	**% VL**	**VL**	**L**	**M**	**H**	**VH**
100	0	0	0.057	0.094	0.189	0.283	0.377
75	25	0	0.060	0.200	0.300	0.340	0.100
75	0	25	0.100	0.200	0.300	0.340	0.060
50	50	0	0.167	0.250	0.250	0.283	0.050

The resulting functions for each of the three main target nodes were automatically calibrated using a genetic algorithm optimized for ranked nodes. For all cases, we reached functions with very low Brier Scores, indicating strong alignment with expert expectations. Table 6 summarizes the learned function types, weights, and average Brier Scores for the training data.

Table 6: Ranked-node function parameters and Brier Scores.

Node	**Function**	**Weights**	**Brier**
Team Fit	WMIN	AT=3, AC=2	0.0028
Tech Fit	WMEAN	AD=4, DE=2, PL=5	0.0048
Collab Fit	WMAX	VL=5, L=4, M=5, H=4, VH=3	0.0070

The BN implemented in Python, all CPTs, elicitation materials, and calibration diagnostics are available in the supplementary material.

4.3 Predictive Accuracy

For the *Team Fit* node, two validation scenarios were tested:

- **Scenario 1**: AT = VH, AC = M → Brier Score = 0.0024
- **Scenario 2**: AT = M, AC = VH → Brier Score = 0.0017

For *Technological Fit*, two scenarios were evaluated:

- **Scenario 1**: AD = VL, DE = VL, PL = M → Brier Score = 0.0015
- **Scenario 2**: AD = VH, DE = M, PL = VH → Brier Score = 0.0008

For *Collaboration Fit*, three validation cases were considered, each reflecting different distributions of pairwise compatibility:

- **Scenario 1**: 50% PC_M, 50% PC_VL → Brier Score = 0.0135
- **Scenario 2**: 25% PC_M, 75% PC_VL → Brier Score = 0.0017
- **Scenario 3**: 75% PC_M, 25% PC_VL → Brier Score = 0.0068

Across all cases, Brier Scores were consistently low, indicating that the model's predictions align closely with expert expectations. These results suggest that the CPTs are well-calibrated and that the BN generalizes effectively to unseen but plausible configurations within the modeled domain.

5 Threats to Validity

This section outlines limitations that may affect the interpretation and generalization of our findings.

Construct Validity. Elicitation relied on a single expert, whose perspective may not capture all dimensions of team fit. The *Collaboration Fit* metric was inferred from project performance records, which may have overlooked interpersonal dynamics not reflected in historical data. **Internal Validity**. While the BN structure and CPTs were reviewed by the expert, no inter-rater checks were performed. Additionally, pairwise compatibility values depend on historical project data, which may contain noise or biases.

External Validity. The study was conducted in a single, process-mature organization. The approach and results may not generalize to other settings with different team structures or data availability. **Conclusion Validity**. Predictive accuracy was evaluated using expert-labeled synthetic scenarios. Although Brier Scores were low, no ground-truth outcome data were available for comparison.

6 Conclusion and Future Work

This study demonstrated the feasibility of modeling expert knowledge for team-project fit using a BN. The approach supports interpretability through a transparent structure aligned with expert reasoning and allows adaptability by enabling integration with broader optimization frameworks for team formation.

Our results indicate that the BN can effectively represent nuanced aspects of developer capabilities, with well-calibrated CPTs and low Brier Scores even on unseen configurations. The use of ranked nodes and structured elicitation enabled precise modeling without overwhelming the expert or requiring extensive data.

Future work will include a comprehensive walkthrough of the full BN with the expert and validation using historical project datasets. We also plan to conduct sensitivity analyses to examine the influence of individual variables, integrate the BN into a genetic algorithm for automated team generation, and explore the generalizability of the model across different organizational contexts.

ACKNOWLEDGMENTS

This work has been partially funded by the project 'iSOP Base: Investigação e desenvolvimento de base arquitetural e tecnológica da Intelligent Sensing Operating Platform (iSOP)' supported by CENTRO DE COMPETÊNCIA EMBRAPII VIRTUS EM HARDWARE INTELIGENTE PARA INDÚSTRIA - VIRTUS-CC, with financial resources from the PPI HardwareBR of the MCTI grant number 055/2023, signed with EMBRAPII.

REFERENCES

[1] Alexandre Costa, Felipe Ramos, Mirko Perkusich, Emanuel Dantas, Ednaldo Dilorenzo, Ferdinandy Chagas, André Meireles, Danyllo Albuquerque, Luiz Silva, Hyggo Almeida, and Angelo Perkusich. 2020. Team Formation in Software Engineering: A Systematic Mapping Study. *IEEE Access* 8 (2020), 145687–145712. https://doi.org/10.1109/ACCESS.2020.3015017

[2] Felipe Cunha, Mirko Perkusich, Danyllo Albuquerque, Kyller Gorgônio, Hyggo Almeida, and Angelo Perkusich. 2025. A Genetic Algorithm with Convex Combination Crossover for Software Team Formation: Integrating Technical and Collaboration Skills. In *Proceedings of the 40th ACM/SIGAPP Symposium on Applied Computing*. 146–153.

[3] Felipe Cunha, Mirko Perkusich, Danyllo Albuquerque, Kyller Gorgônio, Hyggo Almeida, and Angelo Perkusich. 2025. TeamPlus: A data-driven tool utilizing a Genetic Algorithm for optimal software team formation. *SoftwareX* 30 (2025), 102174.

[4] Norman Fenton and Martin Neil. 2018. *Risk assessment and decision analysis with Bayesian networks*. Crc Press.

[5] Norman E Fenton, Martin Neil, and Jose Galan Caballero. 2007. Using ranked nodes to model qualitative judgments in Bayesian networks. *IEEE transactions on knowledge and data engineering* 19, 10 (2007), 1420–1432.

[6] Arthur Freire, Manuel Neto, Mirko Perkusich, Alexandre Costa, Kyller Gorgónio, Hyggo Almeida, and Angelo Perkusich. 2022. A literature-based thematic network to provide a comprehensive understanding of agile teamwork (106). *International Journal of Software Engineering and Knowledge Engineering* 32, 05 (2022), 645–659.

[7] Arthur Freire, Mirko Perkusich, Renata Saraiva, Hyggo Almeida, and Angelo Perkusich. 2018. A Bayesian networks-based approach to assess and improve the teamwork quality of agile teams. *Information and Software Technology* 100 (2018), 119–132.

[8] Gleyser Guimarães, Icaro Costa, Mirko Perkusich, Emilia Mendes, Danilo Santos, Hyggo Almeida, and Angelo Perkusich. 2024. Investigating the relationship between personalities and agile team climate: A replicated study. *Information and Software Technology* 169 (2024), 107407.

[9] Radin Hamidi Rad, Hossein Fani, Ebrahim Bagheri, Mehdi Kargar, Divesh Srivastava, and Jaroslaw Szlichta. 2023. A variational neural architecture for skill-based team formation. *ACM Transactions on Information Systems* 42, 1 (2023), 1–28.

[10] Martin Hoegl and Hans Georg Gemuenden. 2001. Teamwork quality and the success of innovative projects: A theoretical concept and empirical evidence. *Organization science* 12, 4 (2001), 435–449.

[11] Martí Manzano, Claudia Ayala, Cristina Gómez, Antonin Abherve, Xavier Franch, and Emilia Mendes. 2021. A method to estimate software strategic indicators in software development: an industrial application. *Information and Software Technology* 129 (2021), 106433.

[12] George Marsicano, Fabio QB da Silva, Carolyn B Seaman, and Breno Giovanni Adaid-Castro. 2020. The Teamwork Process Antecedents (TPA) questionnaire: developing and validating a comprehensive measure for assessing antecedents of teamwork process quality. *Empirical Software Engineering* 25, 5 (2020), 3928–3976.

[13] Emilia Mendes. 2014. Expert-based knowledge engineering of bayesian networks. In *Practitioner's Knowledge Representation: A Pathway to Improve Software Effort Estimation*. Springer, 73–105.

[14] Maziar Raissi, Zhicheng Wang, Michael S Triantafyllou, and George Em Karniadakis. 2019. Deep learning of vortex-induced vibrations. *Journal of fluid mechanics* 861 (2019), 119–137.

[15] Silja Renooij and Cilia Witteman. 1999. Talking probabilities: communicating probabilistic information with words and numbers. *International Journal of Approximate Reasoning* 22, 3 (1999), 169–194.

[16] Sai Datta Vishnubhotla and Emilia Mendes. 2024. Examining the effect of software professionals' personality & additional capabilities on agile teams' climate. *Journal of Systems and Software* 214 (2024), 112054.

[17] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. 2022. Integrating scientific knowledge with machine learning for engineering and environmental systems. *Comput. Surveys* 55, 4 (2022), 1–37.