A Risk-Integrated Perspective on the Influence of Technical and Environmental Complexity in Software Effort Estimation

Renny Sari Dewi
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia | Department of Digital Business, Universitas Negeri Surabaya, Surabaya, Indonesia
05111960010014@student.its.ac.id

Riyanarto Sarno
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
riyanarto@its.ac.id (corresponding author)

Endang Siti Astuti
Department of Business Administration, Brawijaya University, Malang, Indonesia
endangsiti@ub.ac.id

Received: 27 May 2025 | Revised: 13 July 2025 | Accepted: 1 August 2025

 DOI: https://doi.org/10.48084/etasr.12404

ABSTRACT

Accurate software effort estimation is a persistent challenge in project management, as a significant share of project failures is attributed to inadequate prior risk consideration. Among the widely adopted methodologies that are used for early estimation of the success of a project is the Use Case Points (UCP) method, yet its framework does not take into consideration risk factors, undermining its reliability. To improve upon this limitation, this study introduces UCPRisk, a rule-based expert-guided UCP model that explicitly integrates risk factors across six dimensions: requirements, planning and control, project complexity, user involvement, team dynamics, and organizational environment. The model was evaluated through three public service application case studies, producing effort estimates of 1,564.42 man-hours (agriculture application), 7,287.91 man-hours (company registration), and 4,669.17 man-hours (industrial permit registration). Compared to traditional UCP, UCPRisk reduced the average Relative Error (RE) to 24.3%, demonstrating improved alignment with actual project outcomes. While not a definitive cost calculation tool, UCPRisk provides a structured framework that embeds risk into early estimation, offering more realistic projections for software development.

Keywords-use case points; software risk; risk complexity; software estimation; education quality

I. INTRODUCTION

Concerns regarding the substantial funding required for e-Government application development and the high costs often associated with software solutions are frequently raised by stakeholders; however, these concerns highlight a broader lack of understanding of accurate and reliable software cost estimation. The Project Management Institute (PMI) reported that 27% of project failures in 2017 were attributed to unaddressed risks [1], underscoring the critical influence of risk on software development costs. Several studies have explored this issue, identifying major software risks [2, 3] and establishing links between risk and development effort estimation [4]. Authors in [5], for example, identified ten major software project risks, including unrealistic schedules, inadequate staffing, and changing requirements. Subsequent studies increased risks into six dimensions [6], while the Software Engineering Institute (SEI) [7] introduced a comprehensive software risk taxonomy, outlining 15 critical risk factors commonly encountered in practice.

Furthermore, empirical research has consistently demonstrated a correlation between risk and software development [8, 9]. Several studies have shown that neglecting risk leads to inaccurate estimates of required development effort, increasing the likelihood of cost overruns, schedule delays, and project failure. Despite such evidence, many organizations continue to treat risk analysis as an optional or secondary process, partly because conventional estimation models, such as Function Point Analysis (FPA) and Use Case Points (UCP), do not explicitly incorporate risk factors. Although conducting a comprehensive risk assessment may

initially demand additional time and resources, its absence can result in significant financial losses, reputational harm, and workforce turnover. For these reasons, project managers, software engineers, and clients are increasingly urged to integrate proactive risk mitigation strategies into project planning.

In response and as an extension of prior work in [10], this study seeks to enhance the UCP method by integrating risk parameters defined by two dimensions: probability of occurrence and severity of impact. By embedding these parameters directly into the estimation process, the proposed UCPRisk model aims to generate more realistic predictions of software development effort, which can subsequently be converted into cost estimations. The objective of this research is both theoretical and practical. Theoretically, it demonstrates the value of extending established estimation methods with risk-based parameters, bridging a gap between risk management and software cost estimation research. Practically, it offers project managers and practitioners a more robust algorithm for predicting software effort, reducing the likelihood of underperformance, and promoting more efficient resource utilization.

II. MATERIALS AND METHODS

A. Previous Research

1) Risk Analysis in Software Development

Table I summarizes previous studies that have applied risk analysis techniques in the software development domain. While the study in [11] established foundational weights for each risk dimension using K-means cluster analysis, categorizing them as light, medium, or heavy (Table II), its focus remained on projects emphasizing general business continuity [10]. Our research significantly differentiates itself by analyzing risk parameter observations specifically on public service application development projects. This context shift is crucial because risk severity and impact vary substantially between business-critical and public-facing systems, necessitating a re-validation and re-weighting of the risk parameters in the proposed UCPRisk model.

Although other studies have proposed algorithms for quantifying risk probability [13, 14], their parameters require adaptation and calibration to specific estimation techniques. Consequently, while such studies provide valuable conceptual frameworks, their direct integration into effort estimation models, such as UCP, remains limited. The UCPRisk model addresses this gap by embedding these parameters into the UCP formulation.

Among software effort estimation methods, Constructive Cost Model (COCOMO) [15] explicitly incorporates risk parameters into its analytical structure, while in its updated version, COCOMO II, a Risk Resolution factor is included within the complexity scale computation [16]. Despite several studies advocating the inclusion of risk factors in effort estimation, no existing method directly integrates a comprehensive risk model. An overview of related studies is presented in Table III.

TABLE I. PREVIOUS RESEARCH IN RISK ANALYSIS

Ref.	Method	Result	Parameter Analysis
[5]	RiskMethod	Identified the top-10 software risk items and 18 cost drivers for predicting probability.	Probability and potential loss (impact) are used to produce risk exposure.
[10]	Risk Proportion	Evaluated a risk model on 17 projects by embedding risk in the Technical Complexity Factor (TCF) and Environmental Complexity Factor (ECF).	Technical and environmental risk analysis with risk-based activities.
[11]	K-means cluster analysis	Developed a project risk and performance model with three clusters: low, medium, and high risk.	Six risk dimensions (team, planning & control, user needs, complexity, and organization) are clustered into project risk and project performance.
[12]	Software Cost Estimation, Benchmarking, and Risk Assessment (CoBRA)	Developed a cost estimation model validated on 15 projects using statistical analysis.	Team size, developer skills, project manager skills, requirements quality, reusability, and complexity.

TABLE II. RISK WEIGHT FROM PREVIOUS STUDY

Risk Dimension	Light	Medium	Heavy
Requirements (R)	2.37	3.93	5.36
Planning and Control (P)	2.01	3.50	5.13
Project Complexity (C)	3.58	4.11	4.84
User (U)	2.37	3.56	4.73
Team (T)	2.05	2.29	4.40
Organization (O)	2.75	3.99	5.04
Average Risk Weight (A)	2.52	3.56	4.92

TABLE III. RELATED WORKS IN SOFTWARE EFFORT ESTIMATION

Ref.	Estimation Method	Result	Algorithms/ Tools
[17]	COCOMO II	Calibration of COCOMO II constants using the Fuzzy MOPSO* algorithm reduced MMRE by 11% for effort and 8% for schedule estimation.	Gaussian membership function of fuzzy and MOPSO.
[18]	UCP-Activity Based Costing (UCPabc)	Deviation between UCPabc and actual costs is 2.16%, with an average profit margin of 30.4%.	Activity-Based Costing parameters: cost driver, cost pool, and cost object per activity.

*Multi-Objective Particle Swarm Optimization (MOPSO)

2) Risk-Based Technical and Environmental Complexity in the UCP Method

In the UCP method, complexity is divided into Technical Factors (TF) and Environmental Factors (EF) [17], each rated on a seven-level scale: not recommended, negligible, very low, low, normal, high, and very high (Table IV). The TF and EF defined within the UCP method [19] have traditionally lacked clear categorization in terms of complexity relative to risk

definitions. Table IV introduces a mapping between the six risk dimensions and the existing UCP complexity factors, forming a key theoretical contribution of this study. The proposed model integrates these risk dimensions within the UCP framework by modifying its formulation to explicitly reflect both technical and environmental risk complexity.

TABLE IV. FACTORS CONTRIBUTING TO TECHNICAL AND ENVIRONMENTAL COMPLEXITY

ID	Factors Contributing to Technical Complexity	Level	Risk Dimension
TF1	Distributed system	Very High	Requirements
TF2	Portability	Very High	Requirements
TF3	Application response or throughput performance	Normal	Requirements
TF4	End-user efficiency (online)	Normal	Requirements
TF5	Internal processing complexity	Normal	Project Complexity
TF6	Code reusability	Normal	Project Complexity
TF7	Changeability	Normal	Planning and Control
TF8	Concurrency	Normal	Planning and Control
TF9	Special security features	Normal	Project Complexity
TF10	Provides direct access for third parties	Normal	Planning and Control
TF11	User training facilities	Normal	User
TF12	Easy to install	Low	User
TF13	Easy to operate	Low	User
ID	**Factors Contributing to Environmental Complexity**	**Level**	**Risk Dimension**
EF1	Stable requirements	Very High	Requirements
EF2	Familiar with the method	High	Organization
EF3	Object-Oriented Experience	Normal	Team
EF4	Motivation	Normal	Team
EF5	Application experience	Low	Team
EF6	Analyst capability	Low	Team
EF7	Part-time workers	Not recommended	Organization
EF8	Difficult programming language	Not recommended	Team

B. Proposed Method

The overall research protocol of the proposed UCPRisk is illustrated in Figure 1. The steps followed were:

- Dataset Preparation: The dataset was constructed using case studies from UCP_{ABC} [18] and validated by project managers to ensure data reliability and contextual consistency.
- Risk Clustering: The project risk register was analyzed across all software development phases. Each risk item was clustered based on the logarithmic frequency of occurrence and manually mapped into the corresponding risk categories defined by [11].
- Risk Weighting: Each clustered risk was assigned a weight (light, medium, or heavy) based on the classification from Table II, with adjustments validated by project managers. These weights were then converted into nominal values w for subsequent calculations.
- Probability Assessment: Risk probability p was determined based on its frequency in the risk log. Probabilities were categorized into three levels [2]: i) Improbable: < 0.4, ii) Probable: 0.4-0.7, and iii) Frequent: > 0.7.
- Risk Impact Scoring: The magnitude of potential loss l was evaluated by project managers using the five-level scale presented in Table V, with values ranging from Very Low (1) to Extremely High (5). The risk score RS was then integrated with UCP as shown in (1):

$$UCP_{Risk} = UCP \cdot RS \quad (1)$$

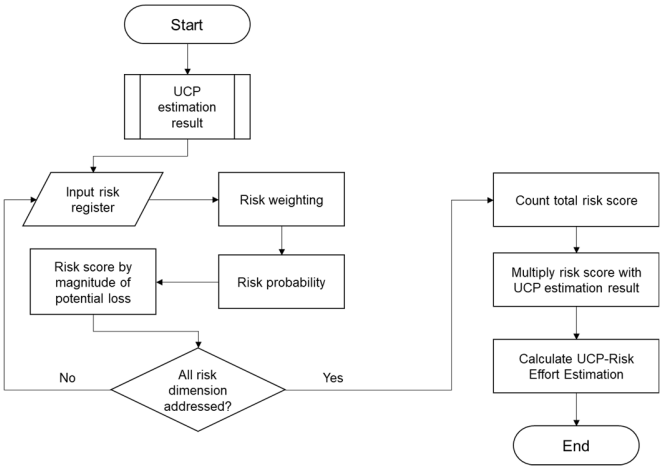

Fig. 1. Research protocol of the proposed UCPRisk.

- Computation of Overall Risk Score: The total risk score was computed by multiplying the magnitude of potential loss by the corresponding weight, assuming a 50% probability (probable) and a moderate impact level (loss magnitude = 3).
- Risk Adjustment in UCP Estimation: The UCP estimation was refined by embedding the computed risk score RS, compensating for deviations observed in Karner's TCF and ECF, which were not equal to 1. The resulting risk correction factor was determined as 0.9892. The refined risk estimation model is expressed as:

$$RS = A - (B \cdot \overline{Cr}) \quad (2)$$

$$\overline{Cr} = \frac{1}{n}\sum_{n=1}^{6} w_n p_n l_n \quad (3)$$

where A is a constant based on the average risk weight, B is a constant value equal to 0.49, $\overline{Cr}$ is the average risk coefficient, w is the weight constant, p is the probability of occurrence, l represents the magnitude of potential loss, and n is the risk parameter.

- Effort Calculation: The final software effort estimation was obtained by multiplying the risk-adjusted UCP value by the productivity rate PR, assumed to be 8.2 man-hours [20], as shown in (4):

$$UCP_{Risk\_Effort\_Estimation} = UCP_{Risk} \cdot PR \quad (4)$$

TABLE V. SOFTWARE RISK IMPACT AND WEIGHT

Potential Loss	Description	Scale
Extremely High	This type of risk is such a natural or man-made disaster that causes a large part of the world to feel its impact. As a result, software development activities cannot be continued.	5
High	Any risk of ethically and materially damaging 60% or more of software development resources.	4
Moderat	If risk happened, it may result in a loss of 40%-60% of software development resources, both morally and materially.	3
Low	The risk is tolerable (between 20 and 40%) and does not have a direct impact on the software development outputs.	2
Very Low	Risks that can be transferred to other parties and are not part of the core business process of software development activities. Therefore, if it occurs, the company will not suffer a material or mental loss of less than 20%.	1

III. RESULTS AND DISCUSSION

A. Collecting Data from Case Studies

The study analyzed three public service application projects, P1, P2, and P3, selected based on [18]. Project P1 was an agricultural application developed by a four-person team. Due to its limited user base and low digitalization level, it was categorized as low risk; most services, such as the distribution of seeds and crops, were still carried out manually. Project P2, a company registration system, was developed by a six-person team. Because company registration processes depend on several prerequisite documents, such as waste licensing, environmental impact analysis, and food feasibility studies, this project was classified as high risk. Project P3, an industry registration application, was developed by a four-person team and involved the registration of small-scale industries with assets below IDR 500 million, with the associated documentation needed being relatively simple; consequently, it was also classified as low risk. Table VI summarizes the main characteristics of these projects.

Building upon prior research in [10], the study extended risk clustering into multiple dimensions, refining the associated weights through integration with other related studies. Project managers also made contextual adjustments to the clustering to reflect the operational reality of each project. The summary of each project's estimation results is shown in Table VII.

TABLE VI. PROJECT CASE STUDIES DETAILS

ID	Project Value (IDR)	Duration (month)	Team (person)	Risk Cluster
P1	46,900,000	2	4	Light
P2	91,500,000	3	6	Heavy
P3	44,300,000	2	4	Light

IRD: Indonesian Rupiah

TABLE VII. PROJECT EFFORT AND RISK CLUSTER

ID	UCP_{ABC} Estimation (IDR)	Average Risk Weight (A)
P1	15,763,743	2.52
P2	46,213,365	3.56
P3	42,440,846	4.92

Table VIII to Table X present the calculated risk coefficients $\overline{Cr}$ based on each risk parameter weight constant w, probability p, and magnitude of potential loss l. Based on these results, the average $\overline{Cr}$ for each of the projects P1, P2, and P3 was 2.87, 6.42, and 2.62, respectively.

Then, using the obtained $\overline{Cr}$ values, the risk score RS was computed using (2), UCP_{Risk} was computed using (1), and $UCP_{Risk\_Effort\_Estimation}$ was computed using (4). The results are summarized in Table XI.

TABLE VIII. RISK COEFFICIENT CALCULATION FOR PROJECT P1

Risk Parameter	Weight Constant w	Probability p	Magnitude of potential loss l	Risk Coef. $\overline{Cr}$
R	2.37	0.3	2	1.42
P	2.01	0.5	2	2.01
C	3.58	0.5	3	5.37
U	2.37	0.5	2	2.37
T	2.05	0.8	2	3.28
O	2.75	0.5	2	2.75

TABLE IX. RISK COEFFICIENT CALCULATION FOR PROJECT P2

Risk Parameter	Weight Constant w	Probability p	Magnitude of potential loss l	Risk Coef. $\overline{Cr}$
R	5.36	0.5	3	8.04
P	5.13	0.5	2	5.13
C	4.84	0.3	3	4.36
U	4.73	0.5	4	9.46
T	4.40	0.3	3	3.96
O	5.04	0.5	3	7.56

TABLE X. RISK COEFFICIENT CALCULATION FOR PROJECT P3

Risk Parameter	Weight Constant w	Probability p	Magnitude of potential loss l	Risk Coef. $\overline{Cr}$
R	2.37	0.3	2	1.42
P	2.01	0.3	2	1.21
C	3.58	0.3	2	2.15
U	2.37	0.5	3	3.56
T	2.05	0.8	2	3.28
O	2.75	0.5	3	4.13

TABLE XI. RESULT OF RISK SCORE AND UCPRISK

ID	Risk Score	UCPRisk	UCPRisk Effort Estimation (man-hours)
P1	1.12	190.78	1,564.42
P2	1.77	888.77	7,287.91
P3	1.24	569.41	4,669.17

To evaluate the effectiveness of the proposed UCPRisk method, the results were compared to estimations obtained using the original UCP method [10]. The original UCP estimates were 171.08, 501.56, and 460.48 for P1, P2, and P3, respectively. Both methods were converted into effort units (man-hours), as shown in Table XII. The discrepancy between the estimated values is then quantified by calculating the Relative Error (RE):

$$RE = \left|\frac{Actual - Expected}{Actual}\right| x\ 100\% \quad (5)$$

where $Actual$ refers to the UCP estimation and $Expected$ refers to the UCPRisk estimation. The results are presented in Table XIII.

TABLE XII. COMPARATIVE RESULT OF SOFTWARE EFFORT ESTIMATION USING UCP AND UCPRISK

ID	UCP	UCPRisk	UCP Effort Estimation (man-hours)	UCPRisk Effort Estimation (man-hours)
P1	171.08	190.78	1,402.86	1,564.42
P2	501.56	888.77	4,112.79	7,287.91
P3	460.48	569.41	3,775.94	4,669.17

TABLE XIII. RELATIVE ERROR MEASUREMENT

ID	RE (%)
P1	10.3
P2	43.6
P3	19.1
Average	24.3

Based on the previous study in [10], the difference between the UCPABC effort estimation and the project contract value, which can be interpreted as gross profit, was 30.4%. Thus, the average RE reported in Table XIII (24.3%) indicates that the company's profit may range from 6.1% to 30.4%, depending on how effectively the project manager handles project risks rather than overlooking them.

In the traditional UCP model, the absence of explicit risk consideration results in uniform scaling of all effort components regardless of contextual uncertainty. Consequently, the estimation deviates considerably when the project involves higher risk exposure, as observed in project P2 where the RE reached 43.6%. In contrast, the UCPRisk model reduces this deviation by embedding risk coefficients that reflect the probability and impact associated with each risk dimension. These results are meaningful not only in terms of numerical comparison but also in their practical implications. UCPRisk enables project managers to anticipate effort deviations earlier in the development process, supporting more realistic and accountable project budgeting, which is particularly important in public service application development.

Lastly, Table XIV summarizes how the proposed UCPRisk model extends and integrates features that were only partially addressed in previous studies. Unlike prior works that focused on risk identification [11], statistical benchmarking [13], or standalone risk assessment [10, 14], the UCPRisk framework provides a more comprehensive treatment of risk factors by accounting for their frequency, weight, and proportional impact on project effort. This cohesive structure strengthens the linkage between risk analysis and software effort estimation, rather than cost estimation alone, and improves the consistency, interpretability, and practical applicability of the estimation process.

TABLE XIV. COMPARISON OF PRIOR RESEARCH AND THIS STUDY

No.	Item Contribution	This Study	Risk Proportion [10]	Risk Cluster [11]	CoBRA [12]	Risk Measurement [14]	RiskMethod [21]
1	Risk frequency.	✓	✓	✓	✓	✓	✓
2	Risk weight.	✓	✓	✓	×	✓	✓
3	Risk proportion.	✓	✓	×	×	×	×
4	Provide a risk model.	✓	✓	✓	✓	✓	×
5	Tested to public service apps development project.	✓	×	✓	×	×	×
6	Embed to software estimation method.	✓	✓	×	✓	×	×
7	Integrate risk into software effort estimation.	✓	✓	×	✓	✓	×
8	Convert risk to software cost estimation.	✓	×	×	✓	×	×
No. of item contributions		8	6	4	7	4	2

B. Threat to Validity and Discussion

While the proposed UCPRisk model demonstrates potential for enhancing early-stage software effort estimation, several threats to its validity must be acknowledged:

- Construct Validity. The model relies on assumptions derived from risk logs and managerial assessments to determine risk scores. Although these assumptions are grounded in empirical observations and supported by prior literature [10, 11], subjective judgments, such as the weighting of risk impact and the categorization of clusters, may introduce bias and affect consistency. To mitigate this, all risk classifications and parameter values should be cross-validated with other experienced project managers involved in similar case studies.
- Internal Validity. The proposed model was evaluated using three real-world case studies involving public service software projects. While a consistent methodology was applied across all cases, the limited number of projects may constrain the causal inferences drawn from the results. Additionally, a notable limitation is the unavailability of the UCP_{ABC} dataset to the public. Although project managers verified the data used in this study, future work could apply the model to open-access datasets to enhance reproducibility and transparency [22].

Additionally, recent studies have increasingly explored the integration of risk into software effort estimation, offering a variety of methodological directions [4, 23-25]. However, a closer examination reveals several conceptual and practical gaps that the proposed UCPRisk model aims to address. For instance, the study in [26] highlights the impact of sequence effects, a form of cognitive bias, on expert-based effort estimation. Their findings suggest that estimation accuracy is not only affected by objective metrics but also by the structure and order of information presented to the estimator. Another research in [27] takes a data-driven approach by applying machine learning to risk assessment in software projects. While their model effectively automates risk classification based on historical project data, its dependence on labeled datasets and prior experience may limit applicability in early-phase or first-time projects, precisely the context in which our UCPRisk model operates. Similarly, the works in [3, 24] attempt to integrate risk variables into effort estimation through comparative analyses of machine learning algorithms. Their studies demonstrate that incorporating risk features can improve estimation accuracy, especially when models are trained on large, representative datasets. However, relying solely on RE as an evaluation metric introduces limitations. These approaches tend to focus on optimization and predictive precision (e.g., using RMSE, MAE, and R^2 metrics), often without accommodating the operational constraints of public service projects, such as the need for explainability, limited data availability, or project-specific risk nuances.

In contrast, the proposed UCPRisk model offers a rule-based, expert-guided framework that explicitly embeds risk exposure into the widely accepted UCP method. It emphasizes transparency, ease of adoption, and contextual adaptability, particularly for public sector environments where standard risk datasets are scarce, and cost justification must be both auditable and understandable to non-technical stakeholders.

IV. CONCLUSION

This study confirms the feasibility of enhancing the Use Case Points (UCP) method through the explicit incorporation of risk parameters. Six dimensions of risk, namely requirements, planning and control, project complexity, user involvement, team dynamics, and organizational environment, have been identified as influential factors in determining software development effort. The proposed UCPRisk integration model was empirically tested across three public service application case studies, producing estimated efforts of 1,564.42 (agriculture application), 7,287.91 (company registration), and 4,669.17 (industrial permit registration) man-hours and yielding an average Relative Error (RE) of 24.3%, indicating improved estimation precision relative to conventional UCP methods.

Importantly, the model is not designed as a definitive tool for precise cost calculation, but rather as a structured framework to support preliminary effort estimation, especially in contexts where detailed historical data is scarce. Its primary strength lies in systematically incorporating risk-related considerations, often overlooked in traditional estimation approaches, into early project planning, thereby promoting more informed and realistic projections in public service application development.

For future work, it is intended to incorporate case studies exhibiting diverse complexities to further investigate the relationship between risk and effort estimation. Additionally, combining UCPRisk with data-driven and hybrid learning approaches may enable the refinement of risk-weight calibration, ultimately supporting more accurate and adaptive software cost estimation methodologies.

ACKNOWLEDGMENT

This work was supported by Institut Teknologi Sepuluh Nopember (ITS) under Riset Kolaborasi Indonesia and Penelitian Departemen.

REFERENCES

[1] PMI Pulse of the Profession, "Success in Disruptive Times: Expanding the Value Delivery Landscape to Address the High Cost of Low Performance," 2018. [Online]. Available: https://www.pmi.org/-/media/pmi/documents/public/pdf/learning/thought-leadership/pulse/pulse-of-the-profession-2018.pdf.

[2] J. Menezes, C. Gusmão, and H. Moura, "Risk factors in software development projects: a systematic literature review," *Software Quality Journal*, vol. 27, no. 3, pp. 1149–1174, Sep. 2019, https://doi.org/10.1007/s11219-018-9427-5.

[3] P. Singal, P. Sharma, and A. C. Kumari, "Risk integrated effort estimation of software projects: a comparative analysis of machine learning techniques," *International Journal of System of Systems Engineering*, vol. 15, no. 2, pp. 166–195, 2025, https://doi.org/10.1504/IJSSE.2025.146198.

[4] R. Natarajan and K. Balachandran, "Ensemble Model of Machine Learning for Integrating Risk in Software Effort Estimation," in *Congress on Intelligent Systems*, vol. 114, M. Saraswat, H. Sharma, K. Balachandran, J. H. Kim, and J. C. Bansal, Eds. Singapore: Springer Nature Singapore, 2022, pp. 635–644.

[5] B. W. Boehm, "Software risk management: principles and practices," *IEEE Software*, vol. 8, no. 1, pp. 32–41, Jan. 1991, https://doi.org/10.1109/52.62930.

[6] L. Wallace, M. Keil, and A. Rai, "How Software Project Risk Affects Project Performance: An Investigation of the Dimensions of Risk and an Exploratory Model*," *Decision Sciences*, vol. 35, no. 2, pp. 289–321, May 2004, https://doi.org/10.1111/j.00117315.2004.02059.x.

[7] M. J. Carr, S. L. Konda, I. Monarch, F. C. Ulrich, and C. F. Walker, "Taxonomy-Based Risk Identification," Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, Technical CMU/SEI-93-TR-6 ESC-TR-93-183, Jun. 1993. [Online]. Available: https://www.sei.cmu.edu/documents/1077/1993_005_001_16166.pdf.

[8] M. N. A. Khan, A. M. Mirza, and I. Saleem, "Software Risk Analysis with the use of Classification Techniques: A Review," *Engineering, Technology & Applied Science Research*, vol. 10, no. 3, pp. 5678–5682, Jun. 2020, https://doi.org/10.48084/etasr.3440.

[9] A. Elzamly, B. Hussin, and N. M. Salleh, "Top Fifty Software Risk Factors and the Best Thirty Risk Management Techniques in Software Development Lifecycle for Successful Software Projects," *International Journal of Hybrid Information Technology*, vol. 9, no. 6, pp. 11–32, Jun. 2016, https://doi.org/10.14257/ijhit.2016.9.6.02.

[10] R. S. Dewi and Y. S. Dharmawan, "A Proposed Model for Embedding Risk Proportion in Software Development Effort Estimation," *Procedia Computer Science*, vol. 234, pp. 1777–1784, 2024, https://doi.org/10.1016/j.procs.2024.03.185.

[11] L. Wallace, M. Keil, and A. Rai, "Understanding software project risk: a cluster analysis," *Information & Management*, vol. 42, no. 1, pp. 115–125, Dec. 2004, https://doi.org/10.1016/j.im.2003.12.007.

[12] A. Trendowicz, *Software Cost Estimation, Benchmarking, and Risk Assessment: The Software Decision-Makers' Guide to Predictable*

Software Development. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[13] C. Kumar and D. K. Yadav, "A Probabilistic Software Risk Assessment and Estimation Model for Software Projects," *Procedia Computer Science*, vol. 54, pp. 353–361, 2015, https://doi.org/10.1016/j.procs.2015.06.041.

[14] J. V. D. Menezes Júnior, "Measuring Risks in Software Development Projects," Ph.D. Thesis, Center of Informatics of Federal University of Pernambuco, Recife, Brazil, 2019.

[15] B. Boehm et al., *COCOMO II Model Definition Manual*. Center for Software Engineering at the University of Southern California (USC), 2000.

[16] B. Boehm, R. Valerdi, and A. Brown, "COCOMO suite methodology and evolution," *CROSSTALK The Journal of Defense Software Engineering*, pp. 20–25, Apr. 2005.

[17] K. Langsari, R. Sarno, and S. Sholiq, "Optimizing Time and Effort Parameters of COCOMO II using Fuzzy Multi-Objective Particle Swarm Optimization," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 16, no. 5, Oct. 2018, Art. no. 2199, https://doi.org/10.12928/telkomnika.v16i5.9698.

[18] R. S. Dewi, G. F. Prassida, Sholiq, and A. P. Subriadi, "UCPabc as an integration model for software cost estimation," in *2016 2nd International Conference on Science in Information Technology (ICSITech)*, Balikpapan, Indonesia, Oct. 2016, pp. 187–192, https://doi.org/10.1109/ICSITech.2016.7852631.

[19] G. Karner, "Resource estimation for objectory projects," *Objective Systems SF AB*, Sep. 1993.

[20] A. Subriadi, S. Sholiq, and P. A. Ningrum, "Critical review of the effort rate value in use case point method for estimating software development effort," *Journal of Theoretical and Applied Information Technology*, vol. 59, no. 3, pp. 735–744, Jan. 2014.

[21] K. Kansala, "Integrating risk assessment with cost estimation," *IEEE Software*, vol. 14, no. 3, pp. 61–67, Jun. 1997, https://doi.org/10.1109/52.589236.

[22] M. Ochodek, J. Nawrocki, and K. Kwarciak, "Simplifying effort estimation based on Use Case Points," *Information and Software Technology*, vol. 53, no. 3, pp. 200–213, Mar. 2011, https://doi.org/10.1016/j.infsof.2010.10.005.

[23] J. T. M. Dhas and Midhunchakravarthy, "The Functional and Storage Risks Associated to the Size Estimation of Parallel Computing Applications," in *Advances in Parallel Computing*, D. J. Hemanth, T. N. Nguyen, J. Indumathi, and S. Lakshmanan, Eds. IOS Press, 2022.

[24] P. Singal, P. Sharma, and A. C. Kumari, "Integrating software effort estimation with risk management," *International Journal of System Assurance Engineering and Management*, vol. 13, no. 5, pp. 2413–2428, Oct. 2022, https://doi.org/10.1007/s13198-022-01652-y.

[25] N. Ramakrishnan, H. A. Girijamma, and K. Balachandran, "Enhanced Process Model and Analysis of Risk Integration in Software effort estimation," in *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, Tirunelveli, India, Nov. 2019, pp. 419–422, https://doi.org/10.1109/ICSSIT46314.2019.8987841.

[26] M. Jørgensen and T. Halkjelsvik, "Sequence effects in the estimation of software development effort," *Journal of Systems and Software*, vol. 159, Jan. 2020, Art. no. 110448, https://doi.org/10.1016/j.jss.2019.110448.

[27] A. Sousa, J. P. Faria, J. Mendes-Moreira, D. Gomes, P. C. Henriques, and R. Graça, "Applying Machine Learning to Risk Assessment in Software Projects," in *Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, vol. 1525, M. Kamp, I. Koprinska, A. Bibal, T. Bouadi, B. Frénay, L. Galárraga, J. Oramas, L. Adilova, G. Graça, et al., Eds. Cham: Springer International Publishing, 2021, pp. 104–118.

AUTHORS PROFILE

Renny Sari Dewi earned a master's degree in informatics in 2014 and continued her career as a lecturer. Since 2021, she has served at Universitas Negeri Surabaya. From 2017 to 2019, she was a member of the Gresik Smart City Council. She is currently pursuing a Ph.D. in computer science, with a focus on software estimation.

Riyanarto Sarno (Senior Member, IEEE) received his Ph.D. in 1992 and is a professor in the Department of Informatics, Institut Teknologi Sepuluh Nopember (ITS). His research interests include machine learning, the Internet of Things, and knowledge engineering. He has conducted research on E-nose technology for five years. He is the author of more than five books and over 300 scientific articles, and was listed among the top 2% of world-ranking scientists in 2020 by Stanford University.

Endang Siti Astuti is a professor at Brawijaya University. She is also a member of the PPIKID Team and a UB Lecturer Certification Assessor. Her research interests include business administration, information system management, perceptions of digital technology, leadership, entrepreneurship, e-commerce, and knowledge management.