engineering proceedings

Proceeding Paper

Diagnostic Framework for Software Engineering Systems: Architectural Design and Validation †

Quang-Vinh Trinh [1,2]

[1] Faculty of Interdisciplinary Science, University of Science, Ho Chi Minh City 70000, Vietnam; trqvinh@hcmus.edu.vn
[2] Vietnam National University, Ho Chi Minh City 70000, Vietnam
† Presented at the 9th Eurasian Conference on Educational Innovation 2026 (ECEI 2026), Da Nang City, Vietnam, 30 January–2 February 2026.

Abstract

A diagnostic framework was developed to evaluate educational projects using performance metrics. The framework adopts an iterative design process in four dimensions: system design, algorithm efficiency, integration complexity, and documentation quality. The framework was tested for 32 projects to assess its scalability, workload balance, and separation between components. Stable performance observed with internal consistency of 0.891 and inter-rater agreement up to 0.775 (Fleiss' Kappa). This framework can be used to evaluate education-related projects for project performance enhancement using educational databases.

Keywords: creative thinking; assessment framework; engineering learning systems; evaluation metrics; design-based research; STEM

1. Introduction

The increasing complexity of distributed computing infrastructures, cloud-native architectures, and intelligent cyber-physical environments in the 4th Industrial Revolution has transformed the evaluation of technological competencies and architectural design strategies. Whereas traditional software engineering evaluation metrics prioritize deterministic, static code metrics, such as line coverage, cyclomatic complexity, or strict procedural adherence, contemporary innovation-driven software evaluations require system-level reasoning, cross-domain technical integration, and adaptive algorithmic problem-solving [1]. As software architecture transitions toward autonomous, modular, and cloud-scale development paradigms, a requirement for automated, modular diagnostic components has emerged to determine non-linear system design variables with high deterministic fidelity [2].

In multi-software environments, integration, deployment, and iterative agile development necessitate multi-dimensional design and architectural patterns [3]. Architectural creativity and divergent design strategies are essential pillars of software performance, driving system-level optimization, runtime adaptability, and long-term architectural sustainability. However, a gap persists between diagnostic and evaluation methods. Most analysis frameworks remain focused on standardized syntactic correctness and deterministic profiling [4–6]. The frameworks cannot capture the multi-dimensional aspects of algorithmic originality and heterogeneous cross-layer integration that modern cloud-native systems demand. This misalignment between optimal design objectives and existing automated evaluation tools restricts the effectiveness of architectural feedback loops. Moreover, the frameworks in software management lack the operational clarity, modular decoupling, and

Check for updates

Academic Editors: Teen-Hang Meen, Chun-Yen Chang and Cheng-Fu Yang

Published: 18 November 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

serialization consistency required for scalable enterprise deployment [7–9]. Conventional cognitive models provide abstract formulations of creativity, but they are inappropriate for integration into automated repository pipelines or data-driven informatics systems [10].

Therefore, a standardized module is needed for non-disruptive diagnostics, structured data serialization, and reliable telemetry across diverse codebases. By developing a diagnostic module, creative innovation in software engineering can be ensured. In this study, we built a design refinement module to diagnose and evaluate system architecture design, algorithmic efficiency, integration complexity, and technical documentation quality [11].

2. Methodology

The diagnostic module was developed using a microservices-based architecture designed for containerized continuous integration/continuous deployment (CI/CD) such as GitHub Actions and GitLab CI/CD. The analytical engine was developed using Python 3.11, FastAPI to manage application programming interfaces (APIs), NetworkX to generate dependency graphs, and abstract syntax tree parsers supporting static code analysis. Data were sorted with Protocol Buffers to reduce latency during repository scans (Figure 1).

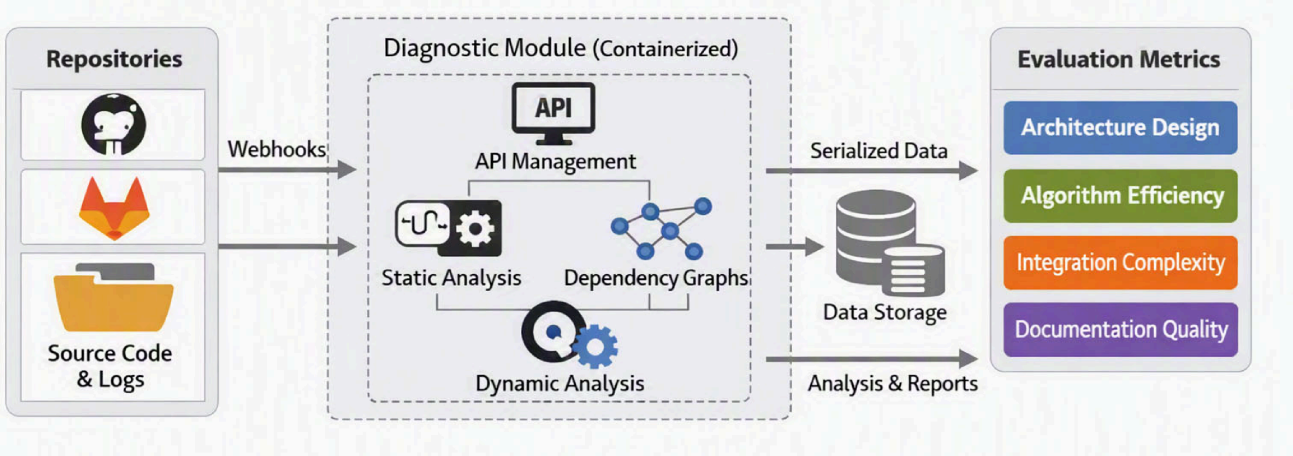


Figure 1. Overview of the diagnostic module (created in this study using AI).

The module performance was validated on cloud-hosted Linux servers (Ubuntu 22.04 LTS) with 8 vCPUs, 32 gigabytes of random-access memory, and non-volatile memory express storage. This device supported parallel processing of the module without performance loss. Repository artifacts, including source code, dependency trees, commit logs, and configuration files, were processed through Git webhooks. The diagnostic module operated for static and dynamic checks and stored performance metrics in postgres object-relational database management system for real-time dashboards.

Three Ph.D. students majoring in software engineering and intelligent systems tested the module for scalability and stability. Then, 18 engineers in Ho Chi Minh City, Vietnam, used the module in diverse runtime environments for performance validation. The module was used to store quality attributes [9,12] to advanced design principles [3,4], producing an evaluation schema with ordinal scoring across four dimensions [11]: system architecture design, algorithmic efficiency, integration complexity, and documentation quality. These evaluation methods were used to compare software quality attributes [9,13] with design principles [3,4], yielding a four-dimensional ordinal scoring schema [11].

- System architecture design was evaluated for divergence from monolithic templates and adherence to modular design patterns;

- Algorithmic efficiency was calculated to assess execution latency, computational complexity O(n), and memory scaling under stress loads;
- Integration complexity was assessed for service orchestration, interface reliability, and cross-layer API interaction quality [11];
- Documentation quality was examined for repository maintainability, inline annotations, API completeness, and clean-code compliance.

The expert panel was formed to review and refine the API definitions, data descriptors, and performance indicators to align the schema with automated deployment standards. Three independent auditors examined the reliability of the results using internal consistency checks (Cronbach's $\alpha \geq 0.80$) and inter-rater agreement (Cohen's and Fleiss' Kappa). Jeffreys's Amazing Statistics Program Version 0.19.3 was used for statistical analysis.

3. Result and Discussion

The diagnostic module was designed around four functional dimensions: originality, problem-solving, interdisciplinary integration, and presentation. Each dimension was evaluated using a four-level ordinal scale (1–4), forming a scoring matrix for assessing creative outputs in engineering contexts. The scale distinguished performance tiers and reduced computational and cognitive overhead, allowing seamless integration into technical practice. To ensure consistency, each criterion was tied to explicit performance descriptors, supporting both formative and summative evaluation across different evaluators. The framework was applied to iterative design cycles and problem-driven engineering tasks, and implemented by 18 participants. The outcomes were evaluated, enabling structured evaluation of creative thinking (Table 1).

Table 1. Pairwise Cohen's Kappa performance across diagnostic dimensions.

Diagnostic Dimension	Mean Cohen's Kappa (κ)	Variance
Interdisciplinary system integration	0.777	Lowest variance; highly deterministic mapping across APIs.
Architectural originality	0.742	Stable distribution; clear differentiation of design patterns.
Technical presentation	0.714	Uniform distribution; minor variance in styling preferences.
Algorithmic problem-solving	0.661	Highest variance; driven by non-linear code strategy tracking.

Internal consistency was measured using Cronbach's α across 32 implementations. The overall reliability coefficient was $\alpha = 0.891$, exceeding accepted thresholds for diagnostic tools [14]. Dimension-specific values ranged from 0.810 to 0.948, confirming strong coherence across constructs. Stability was further assessed using Fleiss' Kappa and Cohen's Kappa. Fleiss' Kappa values ranged from 0.656 to 0.775, indicating substantial agreement [15]. Pairwise Cohen's Kappa values ranged from 0.629 to 0.789, with the highest stability observed in interdisciplinary integration (mean $\kappa = 0.777$) and greater variance in problem-solving (mean $\kappa = 0.661$), reflecting the complexity of evaluating non-linear solution strategies. These results confirm that the module produces consistent, reproducible outputs suitable for standardized diagnostics.

Mean scores showed sensitivity to performance variation: interdisciplinary integration (3.365), originality (3.188), presentation (3.104), and problem-solving (3.094). Low standard errors ($\leq$0.083) indicated minimal variance between evaluators, demonstrating high scoring precision. The framework's sensitivity and stability are comparable to, and in some cases exceed, contemporary measurement systems in engineering education, with structured descriptors reducing rater subjectivity.

The results showed the framework's value as an embedded component of engineering systems. Observational data and stakeholder feedback indicated that it improved the

interpretability of creative performance and supported standardized diagnostic processes. The framework contributed to the standardization of the evaluation of creative outputs across diverse technical contexts, linking learning objectives with quantifiable assessment data and feedback optimization to support closed-loop systems that enable data-informed instructional adaptation.

The framework enables measurable changes in system behavior. A 47% increase in formative feedback frequency and 67% improvement in open-ended design tasks were observed. These results address a long-standing gap in engineering informatics: the absence of structured, integrable tools for evaluating non-linear cognitive performance. Creativity in engineering can be operationalized as a measurable, system-integrated construct rather than treated as an abstract trait. Consequently, the framework can be deployed as a diagnostic component within learning management systems, intelligent tutoring systems, and other data-driven platforms, enabling scalable and modern approaches to engineering education.

4. Conclusions

A diagnostic module was designed to evaluate technical performance, architectural integrity, and code quality in software systems developed by learners. The module applies system design principles to assess system architecture design, algorithmic efficiency, integration complexity, and technical documentation quality. Test results of the module for 32 software projects showed high stability, strong internal consistency (Cronbach's $\alpha = 0.891$), and inter-rater agreement (Fleiss' Kappa of 0.775) with minimal instrumentation overhead. The module can be operated for automated CI/CD. By replacing subjective code reviews with automated assessment, the module can be used to assess software architecture. The module also links development standards to continuous data, making it easy to adopt in enterprise software environments. To overcome limitations of this research, machine learning models must be incorporated to enhance the effectiveness of static code analysis and dynamic execution profiling to support real-time quality monitoring.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Informed consent was obtained from all participants involved in the study. Participation was voluntary, and all collected data were anonymized and used exclusively for research purposes.

Data Availability Statement: The data supporting this study's findings are available from the corresponding author upon reasonable request.

Acknowledgments: The author would like to thank the students who participated in this study and the faculty colleagues who provided valuable feedback throughout the research process. Generative AI (OpenAI ChatGPT, GPT-5.5, and GPT Image 1.5) was used to assist with Figure 1 and language editing. All AI-assisted content was reviewed and verified by the author.

Conflicts of Interest: The author declares that there is no conflict of interest concerning this article's research, authorship, and/or publication.

References

1. Saavedra, A.R.; Opfer, V.D. Learning 21st-century skills requires 21st-century teaching. *Phi Delta Kappan* **2012**, *94*, 8–13. [CrossRef]
2. Bybee, R.W. STEM Education Seems to Be the Answer—What Was the Question? In *The Case of STEM Education: Challenges and Opportunities*; NSTA Press: Virginia, VA, USA, 2013; pp. 41–52.
3. Honey, M.; Pearson, G.; Schweingruber, H. A Descriptive Framework for Integrated STEM Education. In *STEM Integration in K–12 Education: Status, Prospects, and an Agenda for Research*; Honey, M., Pearson, G., Schweingruber, H., Eds.; National Academy of Engineering and National Research Council: Washington, DC, USA, 2014; pp. 31–50.

4. Henriksen, D.; Mishra, P.; Fisser, P. Infusing creativity and technology in 21st-century education: A systemic view for change. *Educ. Technol. Soc.* **2016**, *19*, 27–37.
5. Beghetto, R.A. Does assessment kill student creativity? *Educ. Forum.* **2008**, *69*, 254–263. [CrossRef]
6. Beghetto, R.A.; Kaufman, J.C. Toward a broader conception of creativity: A case for "mini-c" creativity. *Psychol. Aesthet. Creat. Arts* **2007**, *1*, 73–79. [CrossRef]
7. Anaktototy, K. Promoting creativity and innovation in the teaching and learning process. *Edu. Sci. J.* **2023**, *4*, 59–66. [CrossRef]
8. Almeida, L.S.; Prieto, L.P.; Ferrando, M.; Oliveira, E.; Ferrándiz, C. Torrance Test of Creative Thinking: The question of its construct validity. *Think. Ski. Creat.* **2008**, *3*, 53–58. [CrossRef]
9. Sternberg, R.J. Toward a triarchic theory of human intelligence. *Behav. Brain Sci.* **1984**, *7*, 269–315. [CrossRef]
10. Wan, Z.; Zhang, Y.; Xia, X.; Jiang, Y.; Lo, D. Software Architecture in Practice: Challenges and Opportunities. In Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2023), San Francisco, CA, USA, 3–9 December 2023. [CrossRef]
11. Yaroshynskyi, M.; Puchko, I.; Prymushko, A.; Kravtsov, H.; Artemchuk, V. Investigating the Evolution of Resilient Microservice Architectures: A Compatibility-Driven Version Orchestration Approach. *Digital* **2025**, *5*, 27. [CrossRef]
12. Barab, S.; Squire, K. Design-based research: Putting a stake in the ground. *J. Learn. Sci.* **2009**, *13*, 1–14. [CrossRef]
13. Anderson, T.; Shattuck, J. Design-based research: A decade of progress in education research? *Educ. Res.* **2012**, *41*, 16–25. [CrossRef]
14. Tavakol, M.; Dennick, R. Making Sense of Cronbach's Alpha. *Int. J. Med. Educ.* **2011**, *2*, 53–55. [CrossRef]
15. Landis, J.R.; Koch, G.G. The Measurement of Observer Agreement for Categorical Data. *Biometrics* **1977**, *33*, 159–174. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.