World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(REVIEW ARTICLE)

Check for updates

A CONTEXT-AWARE FRAMEWORK FOR INTEGRATING RESEARCH PLANNING, PROJECT MANAGEMENT, AND SOFTWARE ARCHITECTURE DECISION-MAKING: A DESKTOP REVIEW AND FRAMEWORK PROPOSAL

Kosgey Festus Kipchirchir and Samuel Mbugua *

Department of Information Technology, Kibabii University, P.O. Box 1699–50200, Bungoma County, Kenya.
* Corresponding Author

World Journal of Advanced Engineering Technology and Sciences, 2026, 21(02), 010–016

Publication history: Received on 22 September 2026; revised on 01 November 2026; accepted on 03 November 2026

Article DOI: https://doi.org/10.30574/wjaets.2026.21.2.0483

Copyright © 2026 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0

ABSTRACT

The increasing complexity of software systems architecture research requires effective coordination between research planning, project management, and architectural decision-making. These activities influence one another, yet they are frequently addressed through separate methodological and operational processes, so changes in research objectives, technical requirements, resource availability, and project risks may create inconsistencies between planned activities and architectural decisions. This paper examines literature published between 2022 and 2026 to identify approaches, challenges, and research opportunities associated with integrating these activities. Using a desktop-review approach, it synthesizes selected research on architectural decision-making, context-aware systems, software project management, and decision-support mechanisms. The review draws attention to recurring concerns involving decision traceability, stakeholder coordination, contextual adaptation, and limited empirical evaluation of some proposed technical approaches. Based on this synthesis, the paper proposes a context-aware integration framework comprising six components: research objectives and requirements, project planning and control, architectural decision management, context and stakeholder management, risk and uncertainty management, and continuous monitoring and learning. The framework links research objectives to project activities and architectural decisions through traceability mechanisms and context-change triggers. Its proposed contribution is an integrated model for coordinated, evidence-informed decision-making in software architecture research. The framework has not yet been empirically validated, and the desktop review does not constitute an exhaustive systematic-review sample. Future work should implement a reproducible systematic review and evaluate the framework through expert assessment and case studies.

Keywords: Context-Aware Framework; Research Planning; Software Project Management; Software Architecture; Architectural Decision-Making; Framework Evaluation

1 INTRODUCTION

Software systems architecture is concerned with the fundamental structures of software systems, including their components, relationships, interfaces, constraints, and quality attributes. Architectural decisions affect system performance, security, scalability, maintainability, reliability, and adaptability. These decisions are also influenced by business objectives, stakeholder expectations, available resources, technological constraints, and project schedules.

Research planning establishes the objectives, research questions, methodology, activities, resources, and expected contributions of a study. Project management coordinates the implementation of these activities through scheduling, resource allocation, risk management, communication, and progress monitoring. Architectural decision-making determines how technical alternatives are assessed and selected to satisfy system requirements and research objectives. The three activities are closely related. For example, a change in a research question may require a different experimental architecture, additional computing resources, revised milestones, or an alternative evaluation method. Similarly, an architectural decision involving distributed computing or cloud infrastructure may affect project costs, security requirements, implementation time, and the feasibility of the proposed research.

Demir, Chouseinoglou, and Tarhan [1] investigated architectural decision-making through an industrial survey involving 101 practitioners. Their findings highlighted challenges involving business impact, information sharing, and the documentation of decision rationales. These observations illustrate why architectural decisions should be considered within the wider context of project coordination rather than as isolated technical activities. Research on context-aware systems provides another relevant perspective. Santos et al. [2] reviewed context-aware systems architecture in Industry 4.0, identifying interoperability, automation, and decision-making as important themes. Although their work focuses on industrial systems, it provides a foundation for considering how contextual information can support architectural adaptation and coordinated decision-making. These developments motivate the investigation of a context-aware framework that integrates research planning, project management, and software architecture decision-making.

1.1 Statement of the problem

Software systems architecture research involves interconnected technical and managerial activities, including research planning, project scheduling, risk management, and architectural decision-making. When these activities are managed independently, changes in research scope or project conditions may not be effectively communicated, resulting in duplicated work, inconsistent priorities, delayed decisions, inefficient resource allocation, and limited traceability between research objectives and technical outcomes. Although existing literature examines architectural decision-making, context-aware architecture, and software project management, the extent to which these areas can be integrated into a unified framework for software architecture research requires further investigation. This paper therefore addresses the need for a clearly defined, context-aware framework that connects research objectives, project controls, contextual information, and architectural decisions through traceable and adaptable processes.

1.2 Aim of the study

The aim of this study is to develop a conceptual context-aware framework for integrating research planning, project management, and software architecture decision-making based on a desktop review of relevant literature published between 2022 and 2026.

1.3 Research objectives

1.3.1 The study has four objectives:

To examine literature published on research planning, software project management, context-aware systems, and architectural decision-making.

To identify major challenges and contextual factors affecting coordination between research objectives, project activities, and architectural decisions.

To propose a context-aware framework integrating research planning, project management, and software architecture decision-making.

To develop an empirical evaluation approach for assessing the framework's relevance, usability, completeness, and practical applicability.

1.4 Research questions

The paper addresses the following questions:

- What approaches and concepts in the literature are relevant to integrating research planning, project management, and software architecture decision-making?
- What contextual factors and coordination challenges affect these activities?
- How can a context-aware framework integrate research objectives, project controls, and architectural decisions?
- How can the proposed framework be evaluated empirically in software systems architecture research?

1.5 Significance of the study

The paper contributes a conceptual model for coordinating research objectives, project-management activities, and architectural decisions. It provides a basis for linking technical choices to research questions, resource requirements, project milestones, and documented evidence. The framework may be useful to doctoral researchers, software engineering research teams, project managers, software architects, and research supervisors who manage projects involving complex software artifacts, experimental systems, or evolving technical requirements.

2 LITERATURE REVIEW

2.1 Research planning and project management

Research planning establishes the direction and boundaries of a research project. It defines the problem, objectives, research questions, methodology, deliverables, resource requirements, and criteria for evaluating success. Project management translates these plans into coordinated activities. It establishes schedules, responsibilities, dependencies, budgets, communication mechanisms, and risk-management procedures. Related reviews have also examined how software project success is evaluated, including the role of team success criteria [3].

In software architecture research, these activities are particularly important because technical uncertainty can affect the feasibility and execution of the study. A prototype may reveal that an architectural approach does not meet performance requirements, while a change in infrastructure availability may require an alternative experimental design.

Consequently, research planning should be treated as an evolving baseline rather than an entirely fixed document. Changes should be assessed for their implications on research validity, project resources, architectural decisions, and expected outcomes. The integration challenge is to ensure that changes in one project dimension are reflected consistently in the others.

2.2 Software architecture decision-making

Architectural decision-making involves identifying design problems, establishing requirements, generating alternatives, evaluating trade-offs, selecting solutions, and documenting decision rationales.

Demir et al. [1] provide relevant empirical evidence concerning the social and technical dimensions of architectural decision-making. Their survey indicates that decisions frequently involve small groups and collaborative discussions, while information sharing and maintaining decision rationales remain important concerns.

These findings support the need for a framework that records not only the selected architecture but also the reasons for the decision, the alternatives considered, the constraints involved, and the expected consequences.

For research projects, architectural decisions should also be linked to the research objectives they support. This relationship allows researchers to determine whether a technical decision contributes to the intended research outcomes and whether subsequent changes affect the validity or feasibility of the study.

2.3 Context-aware software architecture

Context-aware systems use information about their operating environment to support adaptation and decision-making. Relevant contextual information may include system state, user requirements, resource availability, environmental conditions, organizational priorities, and technological constraints.

Santos et al. [2] reviewed context-aware systems architecture in Industry 4.0 and analyzed 25 articles. Their review identified interoperability, automation, and decision-making as important areas of convergence and highlighted the role

of context representation. Context-awareness applications supported by smart-city infrastructures have likewise been the subject of systematic review [4]. The relevance of this work to research project management lies in the principle that decisions should be evaluated against the conditions in which they are made. A technically appropriate architectural option in one environment may be unsuitable in another because of differences in infrastructure, expertise, cost, security, or time constraints.

This paper extends that principle to research governance by treating context as information that can influence project plans, resource decisions, architectural alternatives, and evaluation procedures.

2.4 Decision-making in software engineering

Decision-making occurs throughout software engineering, from requirements analysis and planning to design, implementation, testing, and operations.

Salin and Rybarczyk [5] reviewed decision-making in agile software engineering, examining decision models, methods, actors, and lifecycle contexts. Their review included 42 studies and identified uneven coverage across lifecycle activities, with greater attention to planning and requirements than to some later stages.

This evidence suggests that decision-making should be understood as a lifecycle process involving multiple actors and activities. For software architecture research, the implication is that decisions need to remain connected to project execution and subsequent evaluation rather than being confined to the initial planning phase. However, agile software engineering is not identical to research project governance. The findings provide relevant background but do not independently establish the effectiveness of a framework integrating all three domains.

2.5 Architectural quality and evidence

Architectural decisions frequently involve trade-offs between quality attributes such as security, performance, maintainability, reliability, and scalability.

The systematic mapping study by Márquez et al. [6] examined architectural tactics and highlighted weaknesses in how some studies define tactics and report their identification methods or data sources. The study also noted limited industrial evidence in the field. These concerns are relevant to research projects because architectural choices should be supported by explicit criteria and appropriate evidence. Researchers need to document the assumptions behind a decision, the methods used to evaluate it, and the evidence supporting the selected alternative. A context-aware integration framework can help connect these technical evaluations to research objectives and project milestones.

2.6 Emerging decision-support technologies

Artificial intelligence, machine learning, and generative AI offer opportunities to support project planning, requirements analysis, risk identification, documentation, and architectural decision-making. Reviews of AI-enabled project management and of AI techniques for cost, effort, and schedule estimation in software projects illustrate the growing attention to such support [7, 8]. Nevertheless, the availability of decision-support tools does not automatically solve coordination problems. Recommendations may be unreliable when contextual information is incomplete, outdated, or poorly represented. Automated suggestions may also overlook research validity, organizational constraints, or stakeholder responsibilities.

AI-based support should therefore be treated as an optional capability within the proposed framework. Its use should be subject to appropriate validation, transparency, human oversight, and documentation.

2.7 Research gap

The reviewed literature addresses several relevant domains, which include architectural decision-making and documentation, context-aware software architecture, project planning and coordination, decision-making across software engineering lifecycles, and architectural quality evaluation. However, the presence of these separate research streams does not demonstrate that they are adequately integrated into a single framework for software systems architecture research.

The proposed research gap is therefore the need to investigate how research objectives, project controls, contextual changes, and architectural decisions can be connected through an explicit and evaluable integration process.

3 RESEARCH METHODOLOGY

3.1 Research design

This paper adopted a desktop-review approach involving the examination and synthesis of accessible scholarly literature. The review informs the development of a conceptual framework and an empirical evaluation protocol.

A full systematic literature review requires a reproducible search strategy, documented eligibility criteria, screening records, quality assessment, and transparent reporting of included studies, as set out in the PRISMA 2020 statement [9]. The present paper does not claim that these procedures have been completed exhaustively.

4 PROPOSED CONTEXT-AWARE INTEGRATION FRAMEWORK

4.1 Framework overview

The proposed framework integrates research planning, project management, and software architecture decision-making through six interrelated components. Research Objectives and Requirements defines the research problem, questions, objectives, methodology, expected contributions, and success criteria that guide the study. Project Planning and Control translates these objectives into manageable tasks, milestones, responsibilities, resource allocations, dependencies, and risk-control measures to support effective implementation. Architectural Decision Management focuses on identifying architectural problems, evaluating alternative solutions, documenting trade-offs, recording decisions, and monitoring their consequences. Context and Stakeholder Management captures changes in research requirements, technological infrastructure, available expertise, stakeholder priorities, organizational constraints, and external obligations that may influence project execution and architectural choices. Risk and Uncertainty Management identifies and assesses technical, methodological, resource-related, scheduling, and operational risks, determining their potential implications for the research process and software architecture. Finally, Monitoring, Evidence, and Learning tracks project milestones, architectural decision status, resource utilization, risks, and evaluation outcomes, ensuring that emerging evidence informs continuous improvements in research planning, project management, and architectural decision-making.

4.2 Framework operation

The framework follows an iterative sequence. First, research objectives and evaluation criteria are established. These objectives are translated into project tasks, milestones, and resource requirements. Architectural decisions are then identified and evaluated against technical requirements, research goals, and project constraints.

The selected decisions are documented together with their assumptions, evidence, owners, and expected consequences. During implementation, the project is monitored for changes in scope, resource availability, technical performance, and risk. When a material change occurs, the framework initiates an impact assessment. The assessment determines whether the project plan, architectural decision, research method, or risk treatment should be revised. Approved changes are documented, and the updated decisions are monitored during subsequent project activities.

5 DISCUSSION

5.1 Integration of research planning and architecture decisions

The framework treats architectural decisions as choices that affect both technical design and research execution. A decision to use a particular architecture may influence the experiments that can be performed, the resources required, and the extent to which the research objectives can be achieved. Explicit traceability makes these relationships easier to examine. It also provides a basis for determining whether a change in architecture requires revisions to the research plan or evaluation method.

5.2 Contextual adaptation and project governance

Contextual changes may affect project feasibility even when the original research objectives remain unchanged. Changes in computing resources, expertise, security requirements, stakeholder expectations, or technical dependencies can alter the suitability of an architectural alternative.

The proposed framework addresses this challenge by incorporating context-change triggers and impact assessments into project governance. Its value will depend on whether relevant contextual information can be captured reliably and acted upon without imposing excessive administrative overhead.

5.3 Implications for software engineering research

The framework has potential implications for three groups. For researchers, it may help maintain a transparent link between research questions, architectural choices, project activities, and reported outcomes, making the reasoning behind a study easier to trace and scrutinize. For project managers, it may make technical dependencies more visible and clarify how architectural changes affect resources and milestones, supporting better planning and risk management. For software architects, it may offer a structured way to bring project constraints, research objectives, and stakeholder priorities into architectural evaluations, so that design decisions reflect all three.

These implications are proposed benefits rather than demonstrated outcomes. Empirical assessment, such as case studies, controlled comparisons, or practitioner evaluations, is needed before they can be reported as improvements.

5.4 Theoretical contribution

The intended theoretical contribution is a model of coordination among research objectives, project controls, architectural decisions, contextual factors, and feedback mechanisms. To establish a stronger doctoral contribution, the study must demonstrate how the proposed relationships or mechanisms differ from existing approaches and whether they explain or improve coordination in ways that established methods do not adequately support. The framework should therefore be treated as a testable research artifact rather than an already validated theory.

6 PROPOSED EMPIRICAL VALIDATION

This desktop-review paper has not yet collected primary data. The following protocol describes how the framework should be evaluated.

6.1 Expert evaluation

The framework may be assessed by software architects, project managers, software engineering researchers, research supervisors, and technical leads. Purposive sampling should identify participants with relevant experience. The sample size should be justified according to the intended analysis and, where inferential testing is planned, an appropriate power analysis.

Participants should assess relevance, completeness, usability, traceability, adaptability, and practical feasibility using a structured questionnaire. Interviews or open-ended questions should identify missing components, implementation barriers, and opportunities for improvement.

6.2 Case-study evaluation

The framework should be applied to selected software architecture research projects or documented project scenarios. Evidence may include project plans, architectural decision records, risk registers, milestone reports, and retrospective interviews, subject to appropriate authorization.

The evaluation should examine whether the framework enables teams to trace architectural decisions to research objectives, identify the implications of context changes, and document adjustments to project plans. Where feasible, comparison with existing project-management practices should be used to assess whether the framework offers measurable benefits.

6.3 Data analysis

Quantitative questionnaire responses may be analyzed using descriptive statistics and suitable reliability assessments. Qualitative responses should be analyzed thematically.

Claims of improved project performance require stronger evidence than expert approval alone. For example, reduced decision delays, improved traceability, fewer unanticipated changes, or more accurate resource planning should be measured using clearly defined indicators and an appropriate comparison design. Ethical approval, informed consent, confidentiality, and secure handling of participant and organizational information should be addressed before data collection.

Limitations

This paper is limited by its desktop-review scope and the absence of a completed, reproducible systematic search. Consequently, the selected literature should not be interpreted as an exhaustive representation of all relevant studies. The framework is conceptual and has not yet been evaluated using expert responses or project-level data. Its practical usefulness, effectiveness, and generalizability therefore remain unconfirmed. In addition, the framework may require

adaptation across different software research domains, organizational settings, and project-management approaches. These limitations should be addressed through a completed systematic review and empirical investigation.

7 CONCLUSION AND RECOMMENDATIONS

This paper proposes a context-aware framework for integrating research planning, project management, and software architecture decision-making. The framework connects research objectives, project controls, architectural alternatives, contextual information, risk management, and continuous monitoring through traceability and feedback mechanisms.

The reviewed literature provides relevant foundations for the proposed integration. Research on architectural decision-making highlights the importance of collaboration and decision documentation, while studies of context-aware systems emphasize the role of contextual information in supporting adaptation and decision-making. The principal contribution is a conceptual model intended to make the relationships between research objectives, project activities, and architectural decisions explicit. However, the originality and effectiveness of the framework remain to be established through further research.

STATEMENTS ON COMPLIANCE WITH ETHICAL STANDARDS

Disclosure of conflict of interest

All authors declare that they have no competing interests.

REFERENCES

[1] Demir MÖ, Chouseinoglou O and Tarhan AK. (2024). Factors affecting architectural decision-making process and challenges in software projects: An industrial survey. Journal of Software: Evolution and Process, 36(10), e2703. https://doi.org/10.1002/smr.2703

[2] Santos A, et al. (2025). Context-aware systems architecture in Industry 4.0: A systematic literature review. Applied Sciences, 15(11), 5863. https://doi.org/10.3390/app15115863

[3] Hans R and Marebane S. (2023). Are software projects evaluated using software teams' success criteria? A systematic literature review. Procedia Computer Science, 219, 1599–1608.

[4] Rocha NP, Dias A, Santinha G and Rodrigues M. (2022). Systematic literature review of context-awareness applications supported by smart cities' infrastructures. SN Applied Sciences, 4, Article 90.

[5] Salin H and Rybarczyk Y. (2026). Decision-making in agile software engineering: A systematic literature review of models, methods, actors and lifecycle contexts. Software, 5(2), 17. https://doi.org/10.3390/software5020017

[6] Márquez G, et al. (2023). Architectural tactics in software architecture: A systematic mapping study. Journal of Systems and Software, 197, 111558. https://doi.org/10.1016/j.jss.2022.111558

[7] Taboada I, Daneshpajouh A, Toledo N and de Vass T. (2023). Artificial intelligence enabled project management: A systematic literature review. Applied Sciences, 13(8), 5014.

[8] Dos Santos RB and Martins LEG. (2026). Artificial intelligence techniques to enhance cost, effort, and schedule estimates in software development projects: A systematic literature review. Journal of Software: Evolution and Process.

[9] Page MJ, McKenzie JE, Bossuyt PM, Boutron I, Hoffmann TC, Mulrow CD, et al. (2021). The PRISMA 2020 statement: An updated guideline for reporting systematic reviews. BMJ, 372,