Online-Journals.org

iJIM International Journal of Interactive Mobile Technologies

iJIM | eISSN: 1865-7923 | Vol. 20 No. 21 (2026) | OPEN ACCESS

https://doi.org/10.3991/ijim.v20i21.63174

PAPER

KCAE: A Knowledge-Centric Architecture for Agile Environments Leveraging LLMs and Knowledge Graphs

Hajar Chnok[1] (✉), Aya Loudiyi Lamghafri[1], Ahmed Chebli Alami[1], Yassine Zaoui Seghroucheni[1], Soumia Ziti[1–3]

[1]Mohammed V University in Rabat, Rabat, Morocco

[2]Hassan II University, Casablanca, Morocco

[3]Moroccan Society for Digital Health, Rabat, Morocco

hajar.chnok@um5r.ac.ma

ABSTRACT

Knowledge management in Agile software development environments remains a critical and underaddressed challenge. Despite the growing body of literature on the subject, existing approaches treat knowledge as a by-product of processes, tools, or human interactions, rather than as a core, explicitly managed system component. This fragmentation limits knowledge reuse, increases individual dependency, and undermines organizational memory across project lifecycles. To address this gap, this paper proposes the knowledge centric architecture for Agile environments (KCAE), a novel architectural framework that places knowledge at the center of both organizational and technical mechanisms. KCAE is organized around four interconnected layers covering the full knowledge lifecycle, leveraging large language models (LLMs), knowledge graphs, graph databases, and retrieval-augmented generation (RAG). Grounded in six design principles and evaluated within the design science research (DSR) paradigm through structured criteria-based assessment and scenario-based validation, KCAE is the first framework to simultaneously achieve high Agile alignment and explicit knowledge lifecycle management. This work opens concrete directions for empirical validation and large-scale deployment in industrial Agile environments.

KEYWORDS

knowledge management, Agile software development, knowledge-centric architecture, knowledge graph, large language models (LLMs), retrieval-augmented generation (RAG), design science research (DSR)

1 INTRODUCTION

Knowledge is increasingly recognized as a key strategic resource for organizations, playing a central role in driving performance, innovation, and long-term competitive advantage [1], [2]. Unlike traditional resources, knowledge is inherently

Chnok, H., Lamghafri, A. L., Alami, A. C., Seghroucheni, Y. Z., Ziti, S. (2026). KCAE: A Knowledge-Centric Architecture for Agile Environments Leveraging LLMs and Knowledge Graphs. *International Journal of Interactive Mobile Technologies (iJIM)*, 20(21), pp. 4–16. https://doi.org/10.3991/ijim.v20i21.63174

Article submitted 2026-08-10. Revision uploaded 2026-09-17. Final acceptance 2026-09-21.

© 2026 by the authors of this article. Published under CC-BY.

dynamic, evolving through continuous interactions between tacit and explicit forms [3]. In this context, the ability of organizations to effectively capture, structure, and leverage knowledge has become a critical capability.

To support these challenges, knowledge management systems (KMS) have been developed to facilitate the collection, organization, sharing, and reuse of knowledge within organizations, with AI-enriched and mobile-enabled KMS increasingly proposed to automate knowledge processing and extend access beyond fixed workstations [4]. Beyond their operational role, such systems contribute to preserving organizational memory and reducing the risks associated with knowledge loss, particularly in environments characterized by workforce mobility and rapidly changing skill requirements [5], [6]. Recent studies further highlight that knowledge-related risks continue to impact organizational performance and sustainability, reinforcing the need for structured knowledge management approaches [7].

These issues are especially relevant in knowledge-intensive domains such as software development, where expertise and system understanding rely heavily on tacit knowledge that is difficult to formalize [8], [9]. They are further amplified in environments adopting Agile methodology, which are characterized by rapid iterations, close collaboration, and continuous knowledge creation. Empirical evidence shows that knowledge management in such environments relies heavily on informal exchanges and collaborative practices, which limit systematic knowledge capitalization and long-term reuse [10].

However, despite the growing use of collaborative tools and practices, existing system architectures remain largely focused on processes, workflows, and tool interactions. In such approaches, knowledge is often treated as a by-product of activities rather than as a core component of the system. As a result, it tends to be fragmented and scattered across multiple tools such as Jira, Confluence, and Git, without a unified structure or explicit lifecycle management. This fragmentation limits knowledge reuse, increases dependency on individuals, and reduces the organization's ability to effectively capitalize on its intellectual assets.

This limitation highlights the need to rethink knowledge management architectures by adopting a knowledge-centric perspective. In this view, knowledge is considered a fundamental, structured, and distributed element of the system, rather than a secondary outcome of processes. In this context, this paper proposes a KMS architecture tailored to Agile environments, aiming to place knowledge at the core of both organizational and technical mechanisms.

2 UNDERSTANDING KNOWLEDGE: FROM DATA TO DECISION-MAKING

The transformation from data to knowledge constitutes a core process in organizational value creation and decision-making. While data and information can be captured and stored relatively easily, the transition toward knowledge is significantly more complex, as it involves interpretation, contextualization, and experiential learning [11]. Knowledge is inherently dynamic and evolves through continuous interactions between tacit and explicit forms [3], making it difficult to fully formalize and structure.

From a strategic perspective, knowledge represents a key organizational asset and a primary source of competitive advantage [1], [2]. Empirical studies further

demonstrate that effective knowledge management has a significant and measurable impact on organizational performance, with reported effect sizes reaching up to 73.9% and strong correlations ranging between 0.73 and 0.83 in relation to employee performance and productivity [12]–[14]. These findings highlight that knowledge is not only a resource but also a critical enabler of organizational intelligence and action.

However, knowledge cannot be reduced to structured information alone. Its contextual, experiential, and partially tacit nature makes it inherently complex, as it is continuously constructed, shared, and transformed through organizational activities [5]. This complexity makes knowledge difficult to capture, transfer, and reuse effectively, particularly when it is embedded within individuals or specific contexts.

This perspective suggests that knowledge should not be viewed merely as a higher level of information, but as a dynamic and context-dependent construct that requires dedicated mechanisms for its structuration and effective utilization. Figure 1 illustrates this transformation and highlights the fundamental characteristics of knowledge within organizational environments.

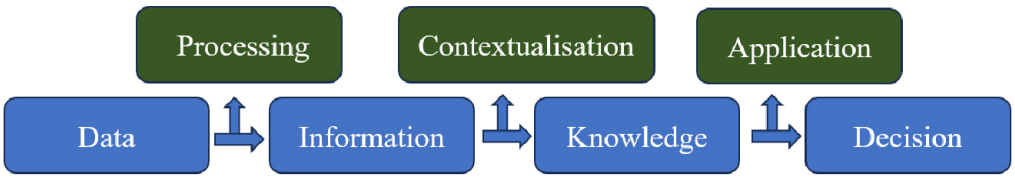


Fig. 1. Knowledge transformation process

3 THE IMPORTANCE OF KNOWLEDGE IN AGILE ENVIRONMENTS

In Agile environments, knowledge constitutes a central and determining element for the effective functioning of development processes. It plays a critical role in rapid decision-making, problem-solving, and continuous adaptation to change. As illustrated in Figure 1, knowledge is part of a transformation process from data to information, then to knowledge, and ultimately to decision-making, making it a direct driver of value creation within projects. Furthermore, Figure 2 highlights the characteristics of Agile environments, including intensive interactions, iterative sprints, rapid decision cycles, and continuous adaptation, which contribute to continuous knowledge creation and sharing.

Unlike traditional approaches, Agile environments rely on continuous and distributed knowledge creation, strongly dependent on human interactions and execution context. Short development cycles, frequent exchanges, and informal communication make knowledge an operational element embedded in daily activities rather than a formally structured artifact [15], [16].

However, this dynamic makes knowledge management particularly complex. Knowledge is largely contextual, evolving, and strongly influenced by individual experience. A significant portion remains tacit and is primarily conveyed through informal interactions. Empirical studies indicate that up to 81% of knowledge management practices in Agile environments rely on direct exchanges, while only a limited portion is formally structured [15]. This situation limits the ability of organizations to effectively structure, capitalize, and reuse knowledge.

Furthermore, the distributed nature of knowledge leads to significant fragmentation, as it is dispersed across individuals, tools, and activities. This increases the risk of knowledge loss and reinforces dependency on individuals [17].

Consequently, although knowledge is a critical asset in Agile environments, its management remains a major challenge. This duality between strategic importance and management complexity highlights the need to rethink existing approaches in order to better structure and leverage knowledge in such contexts.

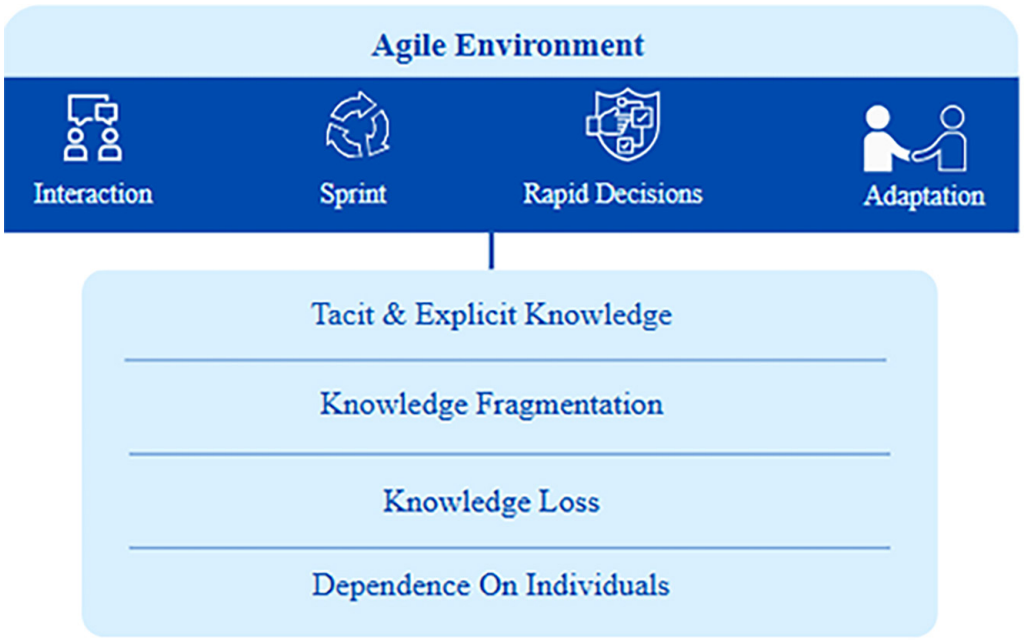


Fig. 2. Knowledge challenges in Agile contexts

4 RESEARCH GAP AND MOTIVATION

While existing literature addresses knowledge management in software development from multiple perspectives, no unified framework has yet been proposed that treats knowledge as a first-class, explicitly managed component within Agile environments.

Architecture knowledge management approaches represent a step toward structuring knowledge artifacts, yet they remain either documentation-heavy, tool-dependent, or misaligned with the iterative and dynamic nature of Agile development [18]. Classical studies on architecture knowledge management further emphasize persistent challenges in capturing, organizing, and reusing architectural knowledge through tools and repositories [19], revealing that these approaches do not fully account for the flexibility and pace required by Agile environments.

From a social and organizational perspective, research on self-organizing roles in Agile teams reveals how informal collaboration and role dynamics implicitly shape knowledge flows within development teams [20]. While this perspective captures the human dimension of Agile work, it provides limited support for structuring or preserving knowledge beyond informal exchanges, increasing dependency on individuals and reducing long-term accessibility.

At a larger scale, empirical studies on very large-scale Agile software development highlight how coordination and process adaptation become increasingly complex as teams grow [21]. In these contexts, knowledge is treated as a by-product of process execution rather than as a managed organizational asset, limiting its explicit lifecycle management and reuse.

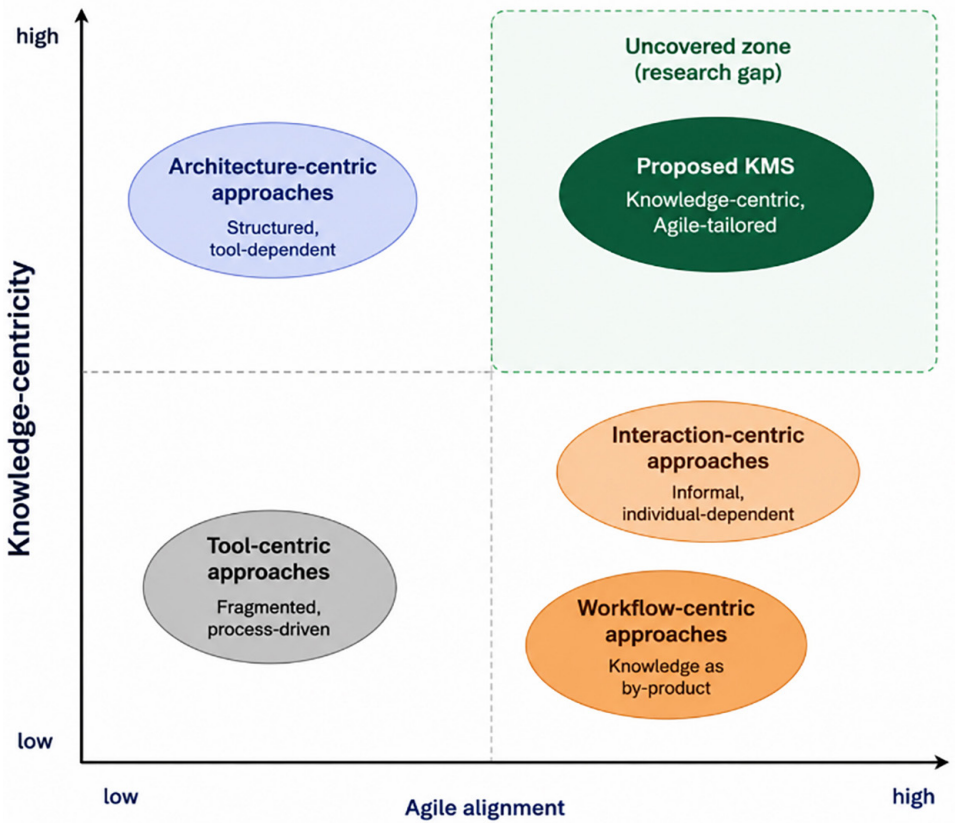


Fig. 3. Positioning of existing literature streams in the knowledge-centricity and Agile alignment space

Collectively, these limitations reveal a consistent pattern across the literature: knowledge is systematically treated as secondary to processes, tools, or human interactions, rather than as a core structural element of the system. As illustrated in Figure 3, no existing approach simultaneously achieves high Agile alignment and explicit knowledge lifecycle management. This gap motivates the present work, which proposes a knowledge-centric KMS architecture specifically designed to address these shortcomings.

5 PROPOSED KNOWLEDGE CENTRIC ARCHITECTURE FOR AGILE ENVIRONMENTS

To address the challenges observed in Agile environments namely knowledge fragmentation, its tacit nature, and its distributed form this paper introduces the knowledge centric architecture for Agile environments (KCAE). Grounded in a knowledge-first perspective, KCAE fundamentally reframes the role of knowledge in Agile systems: rather than emerging as a by-product of activities, knowledge is treated as a primary, explicitly managed entity around which all architectural mechanisms are organized.

Unlike traditional approaches that focus on processes or tools, the proposed KCAE considers knowledge as a central and interconnected entity, linked to Agile activities, interactions, and artifacts. This enables a structured representation of distributed and context-dependent knowledge within Agile environments.

The architecture supports the flow of knowledge from its creation during Agile activities to its structuring, storage, and use in decision-making through a set of interconnected layers.

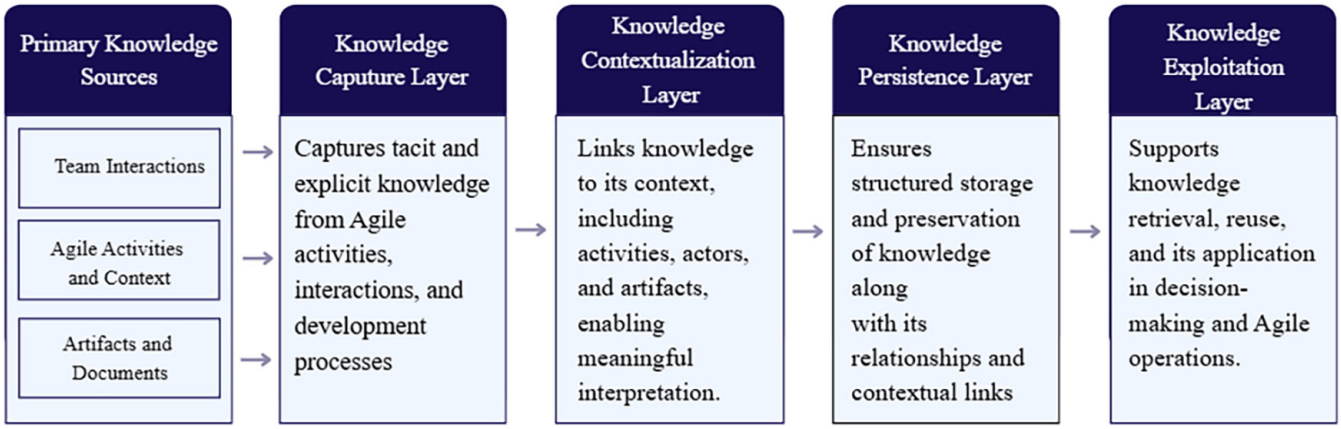


Fig. 4. Knowledge centric architecture for Agile environments

5.1 Knowledge capture layer

Role. Identifies and extracts both tacit and explicit knowledge generated throughout Agile activities, including team interactions, sprint ceremonies, code commits, and development artifacts.

Approach. Large language models (LLMs) are leveraged as automated extraction mechanisms, capable of processing unstructured textual sources and surfacing implicit knowledge that would otherwise remain inaccessible. LLMs serve here not merely as generative tools but as knowledge mediators, transforming informal and distributed information into structured, reusable units [22]–[24]. The detailed evaluation and comparison of extraction models constitute a separate contribution addressed in a companion study.

5.2 Knowledge contextualization layer

Role. Links each captured knowledge unit to its semantic context including its origin, the actors involved, the activities that produced it, and the artifacts it relates to enabling meaningful interpretation beyond raw content.

Approach. Knowledge is structured as a property graph $G = (N, E, \lambda, \varphi)$, where nodes N represent core entities Actors, Artifacts, Activities and typed edges E encode the semantic relationships between them. Knowledge graphs have demonstrated strong capabilities for representing complex, interconnected knowledge structures in software engineering contexts [25]–[27], and their adoption here reflects a deliberate shift from document-centric to relationship-centric knowledge representation.

5.3 Knowledge persistence layer

Role. Ensures durable, structured storage of the contextualized knowledge graph, preserving both its entities and their relational links across the full lifecycle of Agile projects, constituting the organizational memory of the system.

Approach. A graph database engine is adopted as the persistence backbone, chosen for its native ability to store and query deeply connected data structures while maintaining temporal metadata and versioning. This enables knowledge to remain traceable and accessible beyond individual contributions, directly addressing the knowledge loss risk identified in the gap analysis [28].

5.4 Knowledge exploitation layer

Role. Supports knowledge retrieval, reuse, and application in Agile decision-making onboarding, sprint planning, architectural decisions through intelligent, context-aware search.

Approach. Retrieval-augmented generation (RAG) combining the knowledge graph with an LLM to answer queries grounded in organizational knowledge. The integration of RAG with LLMs is rapidly transforming enterprise knowledge management, with hybrid retrieval where dense vectors are complemented by knowledge graphs for structured contexts [29], [30]. In the KCAE architecture, this GraphRAG mechanism transforms passively stored knowledge into an active, intelligent resource directly embedded in Agile workflows.

6 DESIGN PRINCIPLES OF KCAE

The KCAE architecture is grounded in six design principles that guided its conception and justify its structural choices. Each principle emerges directly from the limitations identified in the gap analysis, establishing a clear traceability between the research gap and the proposed solution. Their mapping to the architecture layers is illustrated in Figure 5.

Principle 1: Knowledge-first. Knowledge is treated as a first-class system component rather than a by-product of processes or tools, governing the architecture as a whole [18], [19].

Principle 2: Context-awareness. Every knowledge unit is explicitly linked to its production context actor, activity, and artifact enabling meaningful interpretation and reuse [20].

Principle 3: Lifecycle completeness. The architecture covers the full knowledge lifecycle from capture to exploitation without interruption, ensuring no knowledge is lost between stages.

Principle 4: Agile compatibility. Knowledge management is integrated transparently into Agile practices without imposing documentation overhead, governing both the Capture and Exploitation layers [18], [21].

Principle 5: Structural continuity. Knowledge persists beyond individual contributors, protecting organizational memory against team mobility and role changes [20], [21].

Principle 6: Intelligent accessibility. Knowledge is actively retrievable through context-aware mechanisms, transforming the knowledge base into a resource embedded in Agile decision-making [29], [30].

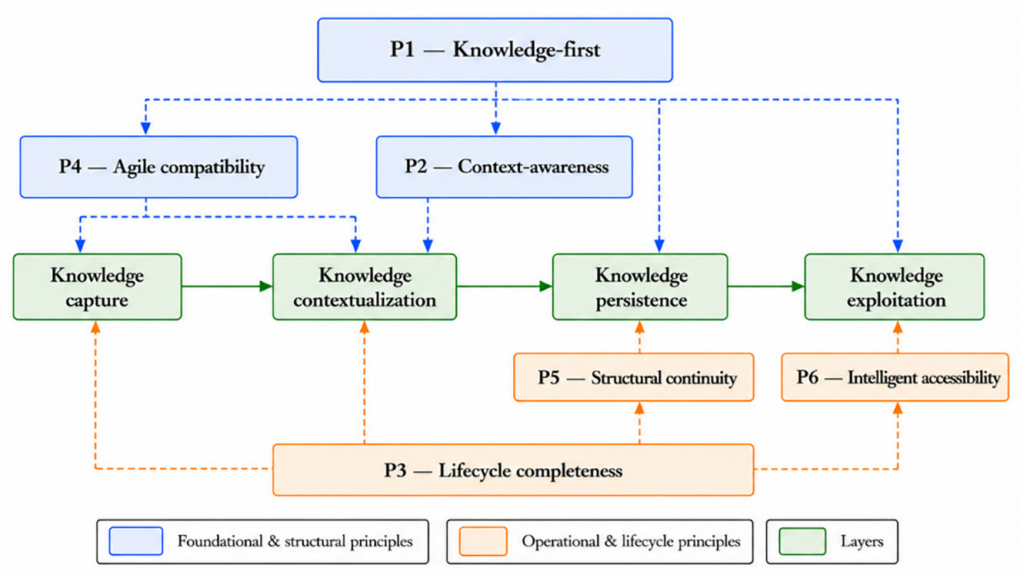


Fig. 5. Mapping of design principles to the KCAE knowledge lifecycle

7 EVALUATION OF KNOWLEDGE CENTRIC ARCHITECTURE FOR AGILE ENVIRONMENTS

7.1 Evaluation methodology

The evaluation of KCAE follows the design science research (DSR) paradigm [31], operationalized through the six-phase methodology [32]. Within this framework, KCAE constitutes a design artifact specifically an architectural model whose validity rests on two complementary dimensions: relevance, established through the gap analysis (Section 4), and rigor, established through theoretical grounding and structured evaluation.

In the absence of a fully operational implementation, we adopt a dual evaluation strategy recognized as valid and rigorous in DSR for conceptual architecture proposals: (1) an analytical evaluation that assesses KCAE against explicitly defined criteria derived from the literature, and (2) an illustrative evaluation through scenario-based demonstration, which shows how the architecture responds to realistic Agile knowledge management challenges. This approach is consistent with formative evaluation strategies, which aim to assess whether an artifact is fit for purpose prior to full deployment.

7.2 Evaluation criteria

To ensure a structured and traceable evaluation, four criteria are derived directly from the limitations identified in the gap analysis and the design principles of KCAE, as summarized in Table 1.

Table 1. Evaluation criteria for KCAE

Criterion	Definition	Derived from
C1 – Knowledge lifecycle coverage	The architecture addresses the full KM lifecycle from capture to exploitation without interruption	P3, gap analysis
C2 – Agile compatibility	The architecture integrates into Agile workflows without imposing documentation overhead	P4, [18], [21]
C3 – Knowledge structuring and context	Knowledge is explicitly structured and linked to its production context	P2, [19], [20]
C4 – Intelligent accessibility	Knowledge is actively retrievable through context-aware, AI-powered mechanisms	P6, [29], [30]

7.3 Scenario-based evaluation

Three scenarios representative of recurring knowledge management challenges in Agile environments are presented. For each scenario, we describe the context, the challenge faced without a structured KMS, the response of KCAE, and the evaluation criteria satisfied.

Scenario 1: Onboarding of a new developer.

Context. A new developer joins an Agile team mid-project. The team is in its sixth sprint and has accumulated significant knowledge about architectural decisions, design rationale, and team conventions across multiple iterations.

Challenge. Without a structured KMS, onboarding relies entirely on informal knowledge transfer through conversations, code reading, and ad hoc documentation. This process is time-consuming, error-prone, and highly dependent on the availability of experienced team members, a well-documented limitation in Agile environments [20], [21].

KCAE response. The Knowledge Capture Layer has continuously extracted and structured knowledge produced during previous sprints, including decisions, artifacts, and interactions, into the knowledge graph. The new developer can query the Knowledge Exploitation Layer using natural language, and the GraphRAG mechanism retrieves contextually grounded answers from the graph, including the actors involved, the activity during which the decision was made, and the artifacts it produced.

Criteria satisfied. C1 (knowledge accumulated across the full lifecycle), C2 (no disruption to existing Agile workflow), C4 (intelligent, context-aware retrieval).

Scenario 2: Knowledge loss after departure of a key team member.

Context. A senior developer who has been central to the project for eighteen months resigns. She carried extensive tacit knowledge about system design choices, technical debt, and workarounds that were never formally documented.

Challenge. In conventional Agile environments, this scenario leads to critical knowledge loss. Tacit knowledge embedded in the individual is not transferable once the person leaves, creating gaps that affect team productivity and decision quality, a risk explicitly identified across the literature [19]–[21].

KCAE response. Throughout the project, the knowledge capture layer continuously extracted both explicit artifacts and tacit knowledge fragments surfaced through the developer's interactions, code commits, and sprint discussions. This knowledge was contextualized in the graph, linked to the relevant activities and decisions, and persisted in the Knowledge Persistence Layer independently of individual contributors. Upon her departure, the knowledge remains fully accessible and traceable through the Exploitation Layer.

Criteria satisfied. C1 (full lifecycle coverage), C3 (knowledge structured and contextualized), C4 (accessible beyond individual contributors).

Scenario 3: Architectural decision during a sprint.

Context. During sprint planning, the team faces a critical architectural decision: whether to introduce an API gateway to manage inter-service communication. A similar decision was debated but not formally documented in sprint 2, eight months earlier.

Challenge. Without a structured KMS, the team has no accessible record of the previous discussion, its rationale, or its outcome. The decision must be made from scratch, risking inconsistency, duplicated effort, and overlooked constraints, a direct consequence of treating knowledge as a by-product of process execution [18], [19].

KCAE response. The team queries the Knowledge Exploitation Layer regarding past discussions about API gateway or inter-service communication. The GraphRAG mechanism traverses the knowledge graph and retrieves the relevant Decision nodes from sprint 2, including the actors who participated, the arguments raised, and the artifacts produced. This enables the team to make an informed, consistent decision in minutes rather than hours.

Criteria satisfied. C1 (knowledge preserved across lifecycle), C2 (seamlessly integrated in sprint planning), C3 (decision contextualized with actors and artifacts), C4 (intelligent retrieval).

8 CONCLUSION

This paper addressed a critical gap in the literature: the absence of a unified architectural framework treating knowledge as a first-class, explicitly managed component in Agile environments, with direct organizational consequences including knowledge loss, individual dependency, and reduced capacity to capitalize on intellectual assets. In response, KCAE was proposed, organizing knowledge management around four interconnected layers and six design principles. The evaluation conducted within the DSR paradigm [31], [32] confirmed that KCAE simultaneously satisfies all four evaluation criteria where no existing approach does.

KCAE provides a structured blueprint for engineering managers and CTOs to rethink how knowledge is captured and preserved across project lifecycles, reducing individual dependency and protecting organizational memory. For Agile teams, the LLM-based capture and GraphRAG-powered retrieval offer a practical path toward embedding knowledge management into existing workflows without documentation overhead, improving onboarding efficiency, decision consistency, and long-term knowledge reuse.

The knowledge graph model, currently defined at a conceptual level, will be further refined through domain-specific ontologies in future work. The LLM-based extraction mechanisms will be comparatively evaluated in a companion study, and the deployment of KCAE in real-world industrial settings will enable empirical assessment of its practical impact.

9 REFERENCES

[1] R. M. Grant, "Toward a knowledge-based theory of the firm," *Strategic Management Journal*, vol. 17, no. S2, pp. 109–122, 1996. https://doi.org/10.1002/smj.4250171110

[2] J. C. Spender, "Making knowledge the basis of a dynamic theory of the firm," *Strategic Management Journal*, vol. 17, no. S2, pp. 45–62, 1996. https://doi.org/10.1002/smj.4250171106

[3] I. Nonaka, "A dynamic theory of organizational knowledge creation," *Organization Science*, vol. 5, no. 1, pp. 14–37, 1994. https://doi.org/10.1287/orsc.5.1.14

[4] H. Prabowo, T. W. Cenggoro, A. Budiarto, A. S. Perbangsa, H. H. Muljo, and B. Pardamean, "Utilizing mobile-based deep learning model for managing video in knowledge management system," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 12, no. 6, pp. 61–73, 2018. https://doi.org/10.3991/ijim.v12i6.8563

[5] L. Argote and P. Ingram, "Knowledge transfer: A basis for competitive advantage in firms," *Organizational Behavior and Human Decision Processes*, vol. 82, no. 1, pp. 150–169, 2000. https://doi.org/10.1006/obhd.2000.2893

[6] D. W. DeLong, *Lost Knowledge: Confronting the Threat of an Aging Workforce*. Oxford, U.K.: Oxford Univ. Press, 2004. https://doi.org/10.1093/acprof:oso/9780195170979.001.0001

[7] T. Dingsøyr and R. Conradi, "A survey of case studies of the use of knowledge management in software engineering," *International Journal of Software Engineering and Knowledge Engineering*, vol. 12, no. 4, pp. 391–414, 2002. https://doi.org/10.1142/S0218194002000962

[8] I. Rus and M. Lindvall, "Knowledge management in software engineering," *IEEE Software*, vol. 19, no. 3, pp. 26–38, 2002. https://doi.org/10.1109/MS.2002.1003450

[9] F. O. Bjørnson and T. Dingsøyr, "Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used," *Information and Software Technology*, vol. 50, no. 11, pp. 1055–1068, 2008. https://doi.org/10.1016/j.infsof.2008.03.006

[10] S. Nerur, R. Mahapatra, and G. Mangalaraj, "Challenges of migrating to agile methodologies," *Communications of the ACM*, vol. 48, no. 5, pp. 72–78, 2005. https://doi.org/10.1145/1060710.1060712

[11] M. Alavi and D. E. Leidner, "Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues," *MIS Quarterly*, vol. 25, no. 1, pp. 107–136, 2001. https://doi.org/10.2307/3250961

[12] M. Shujahat, M. J. Sousa, S. Hussain, F. Nawaz, M. Wang, and M. Umer, "Translating the impact of knowledge management processes into knowledge-based innovation: The neglected and mediating role of knowledge-worker productivity," *Journal of Business Research*, vol. 94, pp. 442–450, 2019. https://doi.org/10.1016/j.jbusres.2017.11.001

[13] R. Masa'deh, B. Y. Obeidat, and A. Tarhini, "A Jordanian empirical study of the associations among transformational leadership, transactional leadership, knowledge sharing, job performance, and firm performance," *Journal of Management Development*, vol. 35, no. 5, pp. 681–705, 2016. https://doi.org/10.1108/JMD-09-2015-0134

[14] V. Y. Isdiani and K. D. Tania, "The influence of knowledge management on employee performance and innovative work behavior at PT. PBT Site Bukit Asam," *Indonesian Interdisciplinary Journal of Sharia Economics*, vol. 8, no. 1, pp. 979–999, 2025. https://e-journal.uac.ac.id/index.php/iijse/article/view/5976

[15] R. A. B. Ouriques, K. Wnuk, T. Gorschek, and R. B. Svensson, "Knowledge management strategies and processes in agile software development: A systematic literature review," *International Journal of Software Engineering and Knowledge Engineering*, vol. 29, no. 3, pp. 345–380, 2019. https://doi.org/10.1142/S0218194019500153

[16] T. Dingsøyr, S. Nerur, V. Balijepally, and N. B. Moe, "A decade of agile methodologies: Towards explaining agile software development," *Journal of Systems and Software*, vol. 85, no. 6, pp. 1213–1221, 2012. https://doi.org/10.1016/j.jss.2012.02.033

[17] B. M. Napoleão, É. F. De Souza, G. A. Ruiz, K. R. Felizardo, G. V. Meinerz, and N. L. Vijaykumar, "Synthesizing researches on knowledge management and agile software development using the meta-ethnography method," *Journal of Systems and Software*, vol. 178, Art. no. 110973, 2021. https://doi.org/10.1016/j.jss.2021.110973

[18] V.-P. Eloranta and K. Koskimies, "Lightweight architecture knowledge management for agile software development," in *Agile Software Architecture: Aligning Agile Processes and Software Architectures*, M. A. Babar, A. W. Brown, and I. Mistrik, Eds. Waltham, MA, USA: Morgan Kaufmann, 2014, ch. 8.

[19] R. Capilla, A. Jansen, A. Tang, P. Avgeriou, and M. A. Babar, "10 years of software architecture knowledge management: Practice and future," *Journal of Systems and Software*, vol. 116, pp. 191–205, 2016. https://doi.org/10.1016/j.jss.2015.08.054

[20] R. Hoda, J. Noble, and S. Marshall, "Self-organizing roles on agile software development teams," *IEEE Transactions on Software Engineering*, vol. 39, no. 3, pp. 422–444, 2013. https://doi.org/10.1109/TSE.2012.30

[21] T. Dingsøyr, N. B. Moe, T. E. Fægri, and E. A. Seim, "Exploring software development at the very large-scale: A revelatory case study and research agenda for agile method adaptation," *Empirical Software Engineering*, vol. 23, no. 1, pp. 490–520, 2018. https://doi.org/10.1007/s10664-017-9524-2

[22] X. Peng, C. Wang, Y. Liu, and W. Zhang, "Code digital twin: A knowledge infrastructure for AI-assisted complex software development," *arXiv preprint arXiv:2503.07967*, 2025. https://arxiv.org/abs/2503.07967

[23] G. Zuin, S. Mastelini, T. Loures, and A. Veloso, "Leveraging large language models for tacit knowledge discovery in organizational contexts," *arXiv preprint arXiv:2507.03811*, 2025. https://arxiv.org/abs/2507.03811

[24] O. Z. Seghroucheni, M. Lazaar, and M. Al Achhab, "Systematic review and framework for AI-driven tacit knowledge conversion methods and machine learning algorithms for ontology-based chatbots in e-learning platforms," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 19, no. 1, pp. 126–139, 2025. https://doi.org/10.3991/ijim.v19i01.51051

[25] L. Wang, C. Sun, C. Zhang, W. Nie, and K. Huang, "Application of knowledge graph in software engineering field: A systematic literature review," *Information and Software Technology*, vol. 164, Art. no. 107327, 2023.

[26] H. Bian, "LLM-empowered knowledge graph construction: A survey," *arXiv preprint arXiv:2510.20345*, 2025. https://arxiv.org/abs/2510.20345

[27] C. Liu and Q. Guo, "A deep learning-driven mobile interaction system for industrial process design: An automotive industry case study," *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 20, no. 8, pp. 172–187, 2026. https://doi.org/10.3991/ijim.v20i08.61247

[28] G. Tamašauskaitė and P. Groth, "Defining a knowledge graph development process through a systematic review," *ACM Transactions on Software Engineering and Methodology*, vol. 32, no. 1, Art. no. 27, pp. 1–40, 2023. https://doi.org/10.1145/3522586

[29] E. Karakurt and A. Akbulut, "Retrieval augmented generation (RAG) and large language models (LLMs) for enterprise knowledge management and document automation: A systematic literature review," *Applied Sciences*, vol. 16, no. 1, Art. no. 368, 2026.

[30] M. Klesel and H. F. Wittmann, "Retrieval-augmented generation (RAG)," *Business & Information Systems Engineering*, vol. 67, no. 4, pp. 551–561, 2025. https://doi.org/10.1007/s12599-025-00945-3

[31] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly*, vol. 28, no. 1, pp. 75–106, 2004. https://doi.org/10.2307/25148625

[32] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, no. 3, pp. 45–77, 2007. https://doi.org/10.2753/MIS0742-1222240302

10 AUTHORS

Hajar Chnok is with the IPSS, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco (E-mail: hajar.chnok@um5r.ac.ma).

Aya Loudiyi Lamghafri is with the IPSS, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco (E-mail: aya.loudiyi@um5r.ac.ma).

Ahmed Chebli Alami is with the IPSS, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco (E-mail: ahmed.chebli@um5r.ac.ma).

Yassine Zaoui Seghroucheni is with the IPSS, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco (E-mail: y.zaoui@um5r.ac.ma).

Soumia Ziti is with the IPSS, Faculty of Sciences, Mohammed V University in Rabat, Rabat, Morocco; Artificial Intelligence and Systems Laboratory, Hassan II University, Casablanca, Morocco; Moroccan Society for Digital Health, Rabat, Morocco (E-mail: s.ziti@um5r.ac.ma).