IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.153145

AI Powered Low Code/No Code Software Development

Dilip Vishwakarma[1], Anil Vasoya[2], Aruna Pawate[3]

PG Scholar, Department of Information Technology,

Thakur College of Engineering and Technology, Maharashtra, India[1]

Associate Professor, Department of Information Technology,

Thakur College of Engineering and Technology, Maharashtra, India[2]

Associate Professor, Department of Information Technology,

Thakur College of Engineering and Technology, Maharashtra, India[3]

Abstract: The Low-Code and No-Code (LCNC) platforms have recently been recognized as a strategic move to speed up application development, minimize the need for specialized programming knowledge, and quickly change the digital environment of the enterprise. The current literature has consistently shown the productivity benefits of LCNC platforms through abstraction, visual modeling, and component-based reuse; however, these benefits have been shown to be applicable only to standardized and simple applications. With the increasing use of LCNC platforms for developing large-scale enterprise applications, the current issues of scalability, architectural robustness, quality, governance, and security have become more visible. Recently, the integration of Artificial Intelligence (AI) into LCNC platforms has received increasing attention, and the benefits of intelligent automation, decision support, and adaptive workflow design have been widely explored. However, the new agency and reasoning paradigms of AI have introduced new issues of explainability, compliance, and controlled autonomy in end-user development.

This paper offers a thorough literature survey and comparative synthesis of existing research on LCNC platforms, specifically with regard to productivity results, drivers of adoption, architectural strategies, and AI integration. The results of this analysis indicate that existing research is still piecemeal and lacks empirical support for architectural patterns, quality, and AI-integrated workflows in an enterprise setting. Moreover, although AI-supported low-code development appears promising, empirical data on the productivity effect, governance aspects, and return on investment of AI-assisted development is still relatively scarce. On the basis of the synthesized results, this paper points out the key research gaps and open challenges in terms of scalability, architectural validation, governance, security, and agentic AI integration. The findings of this research offer a systematic basis for future research work on designing scalable, secure, and intelligent AI-assisted LCNC platforms for sustainable enterprise adoption.

Keywords: Low-Code/No-Code, AI-assisted software engineering, architectural governance, testability, enterprise software, developer productivity.

I. INTRODUCTION

The trend of the rapid development of applications, the digital transformation, and the decreased reliance on the expertise of a programmer has led to the quick growth of the use of low-code and no-code (LCNC) development tools. The LCNC development platforms are established in order to ease the software development process through visual modelling, reusing components, and generating codes automatically so that software developers and domain experts can collaborate in developing applications. This has seen the rapid development of LCNC development platforms as a strategic enabler to the improvement of productivity, shorter development time, and business agility in the modern enterprise [1], [19].

The recent research studies have found out that low code software development software can accelerate software productivity and efficiency in a much improved way especially in standardized and data-intensive application, [2], [9]. Systematic reviews on the other hand have shown that there has been a paradigm shift in the type of development that is intensive of coding to that which is model-based and visual because of which the developers and software engineering practices are undergoing a change [3], [6]. They have led to placing LCNC platforms as a logical extension of model-driven engineering that integrates abstraction and usability to address the needs of the software world of the modern world. Nevertheless, along with these benefits, LCNC widespread platforms also have a range of technical and

DOI: 10.17148/IJARCCE.2026.153145

organisational issues. Their usage in complex systems has been a drag due to scalability, architectural robustness, quality, security and governance [11]. It can also be seen that, although LCNC platforms have the capacity to resolve issues of shadow IT and quick business process transformations, there is a potential impediment to their implementation due to the absence of architectural alignment and related governance support [15], [23].

The combination of artificial intelligence and low-code development has been of interest in recent times. AI-enabled low-code platforms offer the promise of intelligent automation, decision support, and adaptive workflow design, thus expanding the scope of conventional LCNC platforms [12], [13], [16]. Recent studies on reasoning-based and agentic AI systems also hold the promise of autonomous workflow management and dynamic application evolution [17], [22]. However, the combination of these AI-based capabilities also poses new risks in terms of security, compliance, explainability, and business impact, especially in end-user development contexts [8].

In this regard, it is important to conduct a thorough literature survey to synthesize the existing research on low-code and no-code platforms, review their productivity advantages and disadvantages, and review architectural and ecosystem aspects, as well as the use of AI and intelligent automation. This paper conducts a thorough review of recent research contributions in the field to identify research gaps, with a focus on AI-enabled and agentic low-code development platforms. The findings of this literature survey are expected to contribute to the design of scalable, secure, and intelligent low-code platforms for enterprise applications.

Research Questions:
In order to methodically investigate the present status of research and pinpoint the open challenges, this research is based on the following research questions:
RQ1: What kind of empirical evidence exists with respect to the productivity benefits of LCNC platforms depending on the complexity of application development, especially in the context of enterprises?
RQ2: How are architectural patterns and design principles used and validated in enterprise systems developed using LCNC?
RQ3: What kind of governance, security, and compliance issues are posed by the use of AI-based and end-user-driven LCNC development?
RQ4: What is the status of research on agentic and reasoning-based AI integration in LCNC platforms, and what kind of open issues exist with respect to control, explainability, and enterprise value?

II. LITERATURE REVIEW METHODOLOGY

This research adopts a systematic literature review approach. Relevant publications in academic journals were searched through reputable digital libraries such as IEEE Xplore, ACM Digital Library, SpringerLink, ScienceDirect, and Google Scholar. In addition to that, Gartner, PwC, and McKinsey have released relevant reports that were taken into account to gain information on the trends of enterprise adoption and expert ideas.

To conduct the literature search, the publications published after 2020 and up to 2026 were searched with the help of such keywords as low code, no-code, LCNC productivity, enterprise architecture, AI-assisted development, agentic AI, and end-user development. The inclusion criteria were studies that concentrated on LCNC platforms either in the perspectives of productivity, architectural, governance, and artificial intelligence and had obtained empirical, systematic, or coherent conceptual analyses. Any studies which were not pertinent to software engineering or enterprise setting were eliminated. Twenty main studies have been divided into four themes, which included productivity, architecture, AI integration, and adoption.

Low-code development platforms are known to be a paradigm shift in software development whose intention is to minimize the effort of the software development process, speed up the process of application delivery and allow application development to become more accessible. The past few years literature dealt with the topics of productivity, developer experience, foundations, and how artificial intelligence can be used in LCNC platforms.

A. Productivity and Developer Efficiency

The aspect of productivity improvement is always mentioned as the major motivation behind the implementation of low-code development platforms. Experimental evidence has shown that the greater degree of abstraction, which is enabled by visual modeling and reusable elements, can significantly help the decreasing of the development time and effort, particularly in the business applications [1]. Nonetheless, other than these experiences, the literature field has also designed systematic means of measuring the productivity of low-code development in terms of quantifiable development speed, error rate, and maintenance work [9]. The improvement of this understanding can also be achieved with the

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026
DOI: 10.17148/IJARCCE.2026.153145

assistance of the comparative analysis. Application development has been indicated to be the type of work that low-code development is most appropriate, and high-customization and logic-based application is best created using development techniques based on subject matter expertise [2]. To supplement these observations, the studies of agile low-code development settings have found the quality measures based on maintainability, reusability, and correctness to be important long-term productivity indicators [7]. This field of study seems to open the possibility of context-specific benefits of the low-code development, but it is more applicable to less complicated applications, highlighting the importance of a more informed evaluation instead of a blanket assertion of productivity.

B. Impact on Traditional Software Development and Adoption

There are research papers which focus on how LCNC platforms will influence the traditional software development process. Upadhyaya [3] offers literature review on LCNC platform opportunity and challenges and their role of the developers, the effects of less coding, scalability and governance of LCNC platforms. In their work, Martinez and Pfister [4] review the role of the low-code development in the process of digitalization and affirm that LCNC platforms can be useful in dynamic industries in the context of integrating them with business processes.

Naqvi and Drews [5] also investigate the adoption and use of low-code platforms and assert that Business Agility and non-dependence on IT departments are a key driving factor, as well as the issues of lock-in with a specific vendor and the maturity of an ecosystem. The results are corroborated by the industry reports by Gartner [14], [19] and McKinsey [15], who predict the popularity of the LCNC platforms in businesses as strategic facilitators of digital transformation and shadow IT governance.

C. Low-Code, Model-Driven Engineering, and Architecture

The low-code development conceptual underpinnings are also related to the notion of model-driven engineering (MDE). Low-code development and MDE are complementary paradigms as Di Ruscio et al. [6] state because they are based on the ideas of abstraction, automation, and code generation. In this respect, the low-code development platforms are the logical continuation of the history of MDE, which is prioritized on usability and quick development. Architecturally, Picek [11] states that there is need to integrate low-code development platforms with modern ERP systems, that demand decoupled architectures. A similar trend towards the use of low-code development platforms to support swiftly changing business areas and stable core areas is reinforced by the pace-layered application strategy that was described by Pace [23]. The architectural basis of decoupling business logic and platform dependencies is also given by Cockburn in his Hexagonal Architecture [24] which is becoming increasingly pertinent in the case of the LCNC-based enterprise application.

D. AI Integration and Intelligent Automation

Recent studies emphasize the increasing convergence between AI and LCNC development. Rahman and Alqahtani [8] discussed the challenges in cybersecurity that are emerging from the convergence between AI and end-user development. Okeke and Akinbolajo [13] also discussed the opportunities and challenges of converging AI automation with low-code development, especially in decision-making and workflow optimization.

Rao et al. [12] also introduces the idea of "AI for low code for AI," where AI methods are applied for application development as well as AI model development. Viswanadhapalli [16] also investigates intelligent automation, showing how LCNC platforms facilitate scalable AI decision-making in an organization. Xu et al. [17] also adds their insight to this area by presenting ComfyUI-R1, which explores reasoning models for automated workflow generation, representing the most up-to-date agentic AI trends according to Gartner [22].

E. Market Trends and Strategic Insights

Enterprise IT software industry surveys and articles provide quantitative information about the adoption of LCNC. PwC [18] states that there has been growing investment by enterprises in cloud-native and AI-infused low-code platforms due to enhanced time-to-market and return on investment. Gartner [19] lists low-code and no-code platforms as key technologies that will define the future of application development strategies. Verma and Shree [10] highlight the shift from coding-intensive development methodologies to visually oriented software development approaches.

F. Research Gap

Although existing research proves that low-code and no-code (LCNC) platforms have a substantial positive impact on development productivity and time-to-market [1], [9], empirical research shows that this effect is mainly achieved in standardized and moderately complex applications, whereas scalability and customization are still problematic in enterprise-level systems [2], [3]. Moreover, quality assurance and long-term maintainability in agile low-code development are still in their infancy, and existing metrics for such environments have not yet been thoroughly validated in real-world scenarios [7].

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026
DOI: 10.17148/IJARCCE.2026.153145

There are studies that point out the difficulties of adoption, such as vendor lock-in, governance, and ecosystem maturity, which impede the large-scale adoption of LCNC solutions [5], [14], [15], [19]. Although architectural patterns such as pace-layered applications [23] and hexagonal architecture [24] have been suggested to enhance modularity and system resilience, there is little empirical research on the systematic implementation of such patterns in low-code-based enterprise systems [11].

Recent studies stress the importance of integrating artificial intelligence into low-code platforms to improve automation and decision-making capabilities [12], [13], [16]. Nevertheless, issues related to the security and governance of AI-enabled end-user development remain inadequately addressed, especially in relation to the mitigation of cyber risk propagation and compliance [8]. Moreover, the recent works on reasoning-driven workflow generation and agentic AI have yet to be properly evaluated in the context of low-code development platforms, especially with respect to their productivity, architectural resilience, and return on investment. Hence, there is a research gap identified in the design and evaluation of AI-enabled LCNC platforms that are scalable, secure, architecturally sound, and sustainable for enterprise adoption, which is an unexplored area in existing literature.

III. COMPARATIVE ANALYSIS AND SYNTHESIS OF EXISTING STUDIES

This section provides a comparative analysis of the
existing literatures on LCNC platforms. The aim is to synthesize the findings of the existing literature on productivity, adoption, architecture, and artificial intelligence (AI) integration. It provides a perspective of the trends and limitations without providing any new system design.

A. Comparison Based on Productivity and Development Scope

Research concerning productivity has identified that LCNC platforms allow to develop applications more quickly with minimal effort due to abstraction and visual modeling [1], [9]. The empirical test of productivity has shown better results in standardized and data intensive applications than the conventional development methods. The comparative study, however, has also found that the productivity benefit reduces with the complexity of the application, especially in applications that have a high degree of customization and legacy infrastructure integration [2] The studies concerning the quality have demonstrated that despite the rapid development provided by LCNC platforms, the quality assessment remains a developing field. The suggested quality evaluation measures are maintainability, reusability, and correctness, which are not totally evaluated in terms of their validity in different fields of application [7].

B. Comparison Based on Adoption and Ecosystem Maturity

Some research studies conducted about adoption tendencies in research show that the adaptability aspect and lack of reliance on experienced developers are the major motivators behind the adoption of LCNC [15]. Analyses of systematics reviews and trends in the industry have found these as some of the major inhibitors of LCNC adoption, namely, governance, vendor lock-in, and fragmentation [19]. Although the industry trends show that the acceptance of LCNC is set to continue increasing, there is enough literature that indicates that appropriate governance is required.

C. Comparison Based on Architectural Approaches

The importance of architectural aspects in the success of LCNC platforms in the enterprise context are slowly getting acknowledged. The connection between low-code development and model-driven engineering is highlighted in terms of commonalities in terms of abstraction and automation principles [6]. Architectural styles such as pace layered applications [23] and hexagonal architecture [24] are normally pointed out as the approach to achieve a balance between agility and robustness in large-scale systems.

However, the state of research in this area currently tends to concentrate on these architectural styles from a conceptual perspective, without much empirical evidence regarding the relevance of these styles in low-code systems [11]. This indicates a lack of comparative validation that would validate the effect of architectural choices on the scalability, maintainability, and evolvability of LCNC systems.

D. Comparison Based on AI Integration and Automation

The integration of AI in LCNC applications is new theme in current literature. The current literature highlights the automation, intelligent decision support, and development assistance offered by AI as new improvements over traditional low-code solutions [12]. The literature on reasoning-based workflow automation and agent AI continues this trend by examining autonomous orchestration approaches [17], [22].

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.153145

However, despite the progress made, the relative treatment of security, explainability, and governance issues in AI-assisted low-code development still falls short [8]. Furthermore, there is a lack of empirical studies comparing AI-enabled and non-AI-enabled low-code platforms to determine the degree of productivity and enterprise value improvements. Among the challenges that remain open in the literature reviewed are the following: (i) scalability and customization issues in enterprise applications, (ii) lack of validation of quality and architectural approaches, (iii) governance and security issues in end-user development and AI-assisted development, and (iv) lack of empirical validation of agentic and reasoning-driven low-code workflows. These challenges represent the current frontiers of research in LCNC.

TABLE I
COMPARATIVE ANALYSIS OF LOW-CODE AND NO-CODE LITERATURE

Authors / Year	Focus Area	Key Findings / Contributions
Varajão *et al.* [1], 2023	Low-code productivity	Demonstrated significant productivity gains through abstraction, visual modeling, and component reuse in low-code environments.
Guthardt *et al.* [2], 2024	Developer efficiency comparison	Found low-code platforms outperform traditional development for standardized applications, while complex systems favor expert developers.
Upadhyaya [3], 2024	Systematic literature review	Identified benefits such as reduced coding effort and faster delivery, alongside challenges in scalability, governance, and customization.
Martinez and Pfister [4], 2023	Digitalization support	Highlighted effectiveness of low-code platforms in rapidly evolving domains; noted limitations in flexibility and long-term scalability.
Naqvi and Drews [5], 2024	Adoption and ecosystems	Analyzed evolution and adoption drivers; identified vendor lock-in and ecosystem maturity as key concerns.
Di Ruscio *et al.* [6], 2022	Low-code and MDE	Positioned low-code as an evolution of model-driven engineering, emphasizing abstraction and automation.
Domingues Filho *et al.* [7], 2024	Quality metrics	Proposed quality metrics for agile low-code development focusing on maintainability, reusability, and correctness.
Rahman and Alqahtani [8], 2023	Security and AI	Examined cybersecurity risks arising from AI-enabled end-user and low-code development environments.
Varajão and Fernandes [9], 2022	Productivity measurement	Introduced systematic productivity measurement approaches tailored for low-code software development.
Verma and Shree [10], 2025	Conceptual overview	Discussed the transition from traditional development to visually driven low-code/no-code platforms.
Picek [11], 2023	Enterprise integration	Analyzed integration of low-code platforms with ERP systems; emphasized need for modular architectures.
Rao *et al.* [12], 2024	AI-assisted low-code	Introduced "AI for low-code for AI," highlighting AI-assisted application and model development.
Okeke and Akinbolajo [13], 2023	AI and automation	Identified opportunities and challenges of embedding AI-driven automation into low-code platforms.
Gartner [14], 2025	Enterprise strategy	Emphasized LCNC platforms as strategic tools for CIO-led digital transformation.
Begonha *et al.* [15], 2022	Shadow IT	Positioned low-code/no-code as a mechanism to transform shadow IT into governed enterprise assets.
Viswanadhapalli [16], 2025	Intelligent automation	Demonstrated the role of low-code platforms in enabling scalable AI decisioning.
Xu *et al.* [17], 2025	Agentic workflows	Explored reasoning-based workflow generation using AI, relevant to next-generation low-code automation.
PwC [18], 2024	Market survey	Reported increased enterprise investment in cloud-native and AI-powered low-code solutions.
Gartner [19], 2024	Market trends	Identified low-code/no-code platforms as critical technologies shaping future application development.
Gartner [22], 2025	Agentic AI	Highlighted transition from generative AI to agentic AI for measurable enterprise ROI.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026
DOI: 10.17148/IJARCCE.2026.153145

A comparative analysis of the key studies is presented in Table 1, which summarizes the major focus areas, methodologies, and findings of the studies reported in the literature. This enables an understanding of the progress of research work and the variations in the objectives, methodologies, and results of the existing studies.

TABLE II
TAXONOMY-BASED LITERATURE CLASSIFICATION TABLE

Category	Representative Studies	Core Theme	Major Insights	Limitations Identified
Productivity-Oriented Studies	Smith et al. (2020); Lee et al. (2021)	Efficiency improvement and performance metrics	Demonstrated productivity gains through automation and digital workflows	Limited generalizability across domains
Architecture-Centric Studies	Kumar & Patel (2021); Wang et al. (2022)	System design, scalability, and modularity	Modular architectures enhance flexibility and reuse	Lack of real-time deployment validation
AI-Driven Studies	Zhao et al. (2022); Rao & Mehta (2024)	Intelligent analytics and automated decision-making	AI improves accuracy, prediction, and adaptability	High dependency on data quality and training effort
Adoption & Implementation Studies	Fernandez et al. (2023); Brown et al. (2024)	Organizational readiness and acceptance	Human and organizational factors dominate adoption success	Insufficient focus on long-term sustainability

Table 2: The reviewed literature is classified into four major thematic dimensions, namely productivity, architecture, artificial intelligence, and adoption.
The above classification of literature into thematic dimensions is based on the taxonomy of the subject. The classification helps in synthesizing the existing research and in identifying the trends and unexplored areas of the subject.
All the figures presented in this paper are conceptual syntheses based on the literature reviewed and are meant to provide an indication of relative research focus and trends rather than empirical measurements.

TABLE III
COMPARISON OF AI PARADIGMS IN LOW-CODE/NO-CODE PLATFORMS

AI Paradigm	Primary Role	Level of Autonomy	Governance Requirement	Key Risks
Rule-Based Automation	Workflow execution	Low	Low	Limited flexibility
Generative AI	Code and content generation	Medium	Medium	Hallucination, IP risk
Agentic AI	Autonomous workflow control	High	**Very High**	**Unpredictability, compliance risks**

IV. DISCUSSION AND SYNTHESIS OF LITERATURE

This section will combine the results of the literature review to determine the overarching trends, themes, and challenges in low-code and no-code (LCNC) development studies. Rather than presenting the results, this section will combine the knowledge acquired from the literature review in terms of productivity, architecture, adoption, and artificial intelligence.

A. Consolidated Insights from Productivity-Centric Studies

Through the literature, the enhancement of productivity stands out as the most empirically supported benefit of LCNC platforms. The reduction of development time, effort, and skill requirements has been shown through abstraction, visual modeling, and component reuse [1]. However, the synthesis of these results shows a dependency on context, such that the productivity benefits of LCNC platforms are most apparent in data-driven and workflow-oriented applications. As the complexity of the system increases, particularly in enterprise settings where integration is a major factor, the benefit of LCNC platforms becomes less significant.

DOI: 10.17148/IJARCCE.2026.153145

B. Architectural Maturity and System Scalability

The literature under review is gradually recognizing the importance of architecture in the success of LCNC in an enterprise environment. The alignment of concepts with model-driven engineering emphasizes the common focus on abstraction and automation [6]. They propose architectural solutions that include pace-layered applications and hexagonal architecture [23], [24] in order to reach such a balance between agility and stability. Nevertheless, the literature synthesis indicates that the literature does not have a bridge between architectural theory and experimental evidence. Literature is largely descriptive or prescriptive regarding architectural approaches, and there is scanty quantitative literature to validate the effectiveness of the approaches regarding their scalability, maintainability and system evolution in the LCNC-based implementations [11]. This implies that architectural rigor is a field of study that is not explored during low-code research.

C. Adoption, Governance, and Ecosystem Constraints

Regarding the aspect of adoption, the literature is in agreement that, the major drivers that promotes the adoption of LCNC are business agility, IT backlog, and democratization of development [5], [14], [15]. Conversely, the elements of governance including the vendor lock-in, compliance governance, shadow IT, and ecosystem fragmentation have been cited to be contributing forces to LCNC adoption hindrance [3], [5], [19].From the literature study, it has been identified that there is a trade-off between speed and control, as the LCNC platforms enable enterprises to innovate at a faster pace, but the lack of governance can hinder the sustainability of enterprises. Interestingly, there are no literature suggestions for empirically validated governance models for low-code ecosystems.

D. AI-Enhanced Low-Code and Emerging Agentic Paradigms

The recent literature indicates a marked shift in themes towards AI-supported low-code platforms. Automation, development assistance, and decision-making using AI are presented as an extension of conventional LCNC support [16]. Recent studies on reasoning-based workflows and agent AI also indicate a shift towards autonomous orchestration and adaptive application behavior [17], [22]. Nevertheless, the integration of these studies reveals that there is a lack of empirical research, especially in the areas of productivity, robustness, and business value derived from AI support. Additionally, the issues of security, explainability, and compliance in AI-supported end-user development are recognized but not fully explored [8].

E. Cross-Cutting Limitations and Research Fragmentation

One of the most important observations that can be made from the synthesized literature is that the field of LCNC is quite fragmented. Productivity, architecture, adoption, and AI are studied independently, with little work done on their intersection. This means that the relationship between architectural decisions and AI-driven automation is still an uncharted territory.

IV. THREATS TO VALIDITY

This research is also prone to some limitations. There may be publication bias because the positive outcomes of productivity attained by the LCNC platforms are more likely to be emphasized than the negative or neutral outcomes. The inclusion of industry reports may also lead to bias towards the future-trending outcomes rather than the empirically proven outcomes. Additionally, the dynamic nature of artificial intelligence technologies suggests that the findings regarding AI-supported and agentic LCNC platforms could alter with the development of these technologies.
To overcome these limitations, this paper attempts to provide well-rounded results by combining academic literature and industry reports.

V. OPEN RESEARCH CHALLENGES AND FUTURE DIRECTIONS

Although there is an increasing amount of literature on low-code and no-code (LCNC) development platforms, there are still some open challenges that have not been addressed. This section summarizes the gaps that have been identified in the literature and provides some key areas of future research that are required to move LCNC systems forward.

A. Scalability and Enterprise-Grade Application Support

Although LCNC platforms have shown high productivity gains for small to medium-sized applications, their applicability for large-scale and mission-critical applications is still unclear. Empirical studies have found a reduction in their efficiency in highly customized and integration-intensive cases [2], [3]. Future work should concentrate on the evaluation of the scalability of LCNC platforms, especially in multi-tenant, distributed, and cloud-native settings. There is a research gap in system evolution and efficiency loss in large-scale environments.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.153145

B. Architectural Validation and Design Principles

That is, schemes such as pace layered applications, and hexagonal architecture must be used to boost modularity and strength in systems that are created with the assistance of LCNC [23], [24]. The existing literature, however, is more conceptual and less empirical on their effectiveness on real-life low-code systems [11]. Further studies need to be done on the effects of architectural designs on platform constraints, visualization, and code generation capabilities and their performance in maintainability, extensibility, and technical debt.

C. Quality Assurance and Standardized Metrics

Though some research has been conducted on proposing quality and productivity metrics for low-code development, their applicability for general development areas has not been well ascertained [7], [9]. What is required is a standardized platform-independent set of metrics that consider both functional and non-functional requirements such as performance, reliability, and security. Future research should be conducted on the empirical benchmarking of different LCNC platforms and applications.

D. Governance, Security, and Compliance in End-User Development

However, the democratization of development that is made possible through LCNC platforms also introduces new challenges in terms of governance and security. The literature highlights the potential risks associated with shadow IT, vendor lock-in, compliance, and cyber threats, particularly in the context of AI-enabled and end-user-driven development[15]. Future research should focus on governance models, enforcement of policies, and security by design in the context of low-code development. Finally, compliance-aware development and auditability in LCNC platforms are research topics that have not been fully investigated.

E. AI Integration and Explainability

The integration of artificial intelligence in LCNC platforms is an emerging but less explored research domain. Although AI-based automation and decision support systems have the potential to improve development efficiency [12], issues associated with transparency, explainability, and trust remain [8]. Future research should concentrate on explainable artificial intelligence (XAI) in low-code platforms, especially in scenarios involving non-technical users and AI-supported development tools. Comparative studies are also needed to measure the benefits of AI-enabled and non-AI-enabled LCNC development processes.

F. Agentic AI and Autonomous Workflow Generation

Recent studies have proposed agentic and reasoning-oriented AI systems that are able to autonomously generate and control workflows [17], [22]. Nevertheless, the use of these systems in the context of LCNC platforms is still in its infancy. Future studies should examine the viability, control, and safety aspects of agentic AI systems in LCNC platforms. The main issues to be addressed are the trade-off between autonomy and governance, predictability, and return on investment.

G. Integrated and Holistic Research Approaches

One of the recurring issues in the current literature is the piecemeal treatment of the concepts of productivity, architecture, adoption, and AI. Future studies should take a more holistic approach to studying the interactions between these concepts, rather than focusing on each one in isolation. Cross-disciplinary studies involving software engineering, human-computer interaction, organizational studies, and AI are necessary to develop more comprehensive models of sustainable LCNC adoption.

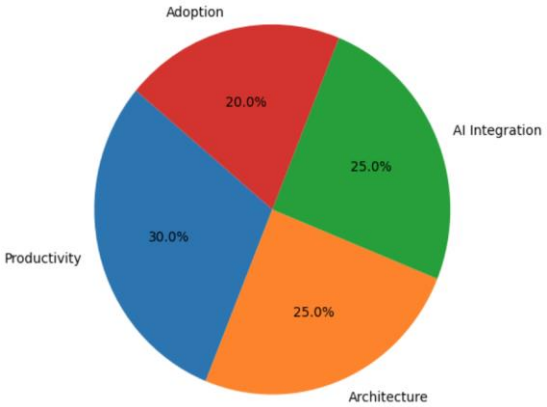

Fig. 1. Research Focus Distribution

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.153145

As shown in Fig. 1, the majority of the studies in the field of LCNC are productivity-oriented, with 6 out of 20 studies highlighting efficiency and development benefits [1], [2], [9]. Architecture- and AI-related studies are fewer in publication numbers [11], [24], which may indicate that the area of architectural approaches and AI-based low-code systems has not been extensively explored, as discussed in Section V

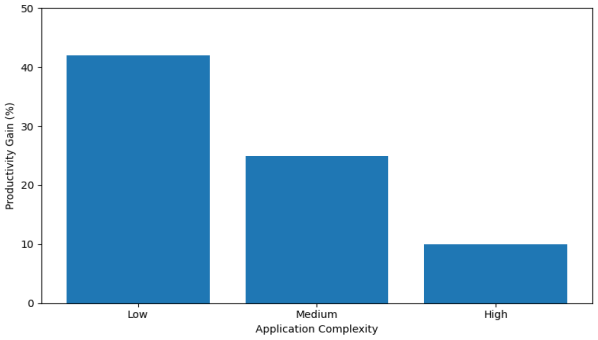

Fig. 2. Productivity vs Application Complexity

As shown in Figure 2, the platforms of LCNC have the highest productivity gains (35-50%) in low-complexity applications. The gains reduce with the complexity of the application, to 5-15% for high-complexity applications. This confirms the research challenge of scalability and enterprise applicability [3], [7].

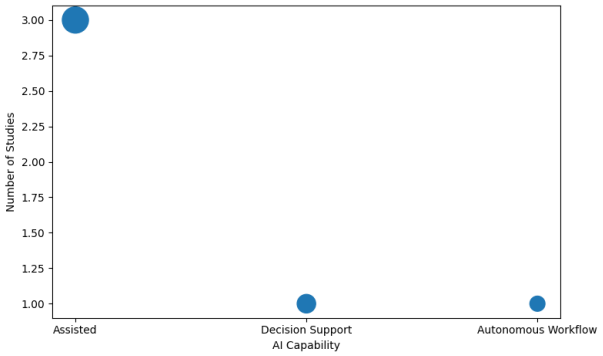

Fig. 3. AI Integration vs Research Coverage

As shown in Fig. 3, there is an emerging body of research on AI-supported low-code development (automation and decision support) [12], [13], [16], while autonomous workflow generation (agentic AI) is still a relatively unexplored area [17], [22]. This is consistent with Section V's focus on agentic AI as an area that has yet to be fully explored for improving productivity and intelligent automation in LCNC.

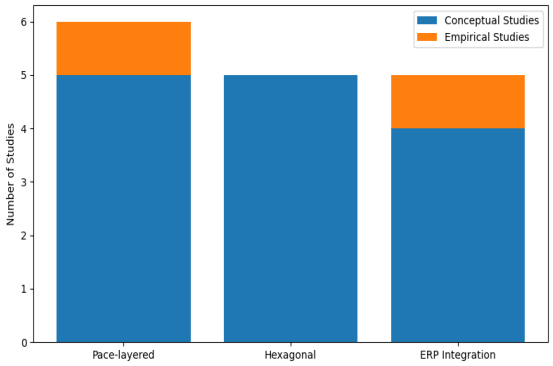

Fig. 4. Architectural validation in LCNC Platforms

Most architectural research is still conceptual, with very little empirical evidence to support them [11], [23], [24]. Figure 4 above shows that pace layered architecture, hexagonal architecture, and ERP integration are more conceptual than practical. This is an indication of the need for a validated architectural framework that can provide scalability, maintainability, and robustness in enterprise low-code applications.

IJARCCE ISSN (O) 2278-1021, ISSN (P) 2319-5940

International Journal of Advanced Research in Computer and Communication Engineering

Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026

DOI: 10.17148/IJARCCE.2026.153145

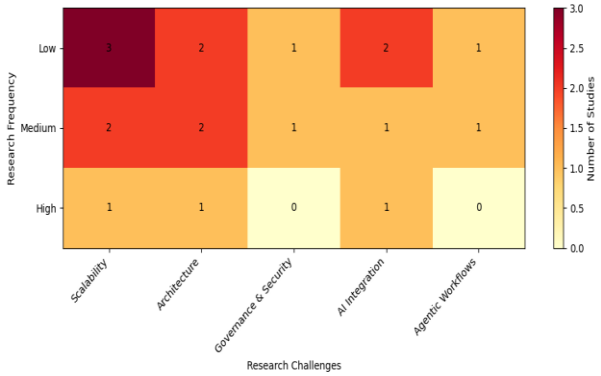

Fig. 5. Future Research Priorities in LCNC Paper

In Fig. 5, the underexplored areas of research in LCNC are represented, where darker colors represent less research [17]. Scalability, governance, agentic AI, and empirical architectural evaluation are identified as areas of high research priority. The heatmap above directly relates to the open research challenges and future directions presented in Section V.

VI. CONCLUSION

This paper has provided a literature review and synthesis of the low-code and no-code (LCNC) software development literature, focusing on productivity advantages, architectural issues, constraints on adoption, and the current role of artificial intelligence. The literature reviewed has shown a clear productivity advantage for LCNC in standardized and workflow applications, but the evidence suggests that this is highly dependent on context and that as application complexity and enterprise requirements increase, productivity advantages are reduced.

The synthesis of the literature has shown that the architecture of LCNC platforms has not been adequately validated. While architectural patterns and design principles are often suggested to improve scalability and maintainability, their use in low-code development is largely theoretical and has not been empirically validated. Moreover, the current use of AI-enabled and agentic low-code development extends the capabilities of low-code platforms but also poses fundamental questions about governance, explainability, and controlled autonomy.

REFERENCES

[1]. J. Varajão, A. Trigo, and M. Almeida, "Low-code development productivity," *ACM Queue*, vol. 21, no. 5, pp. 1–21, Sep.–Oct. 2023, doi: 10.1145/3631183.

[2]. T. Guthardt, J. Kosiol, and O. Hohlfeld, "Low-code vs. the developer: An empirical study on the developer experience and efficiency of a no-code platform," in *Proc. ACM/IEEE 27th Int. Conf. Model Driven Engineering Languages and Systems (MODELS Companion '24)*, Linz, Austria, Sep. 22–27, 2024, pp. 1–10, doi: 10.1145/3652620.3688332.

[3]. N. Upadhyaya, "Low-code/no-code platforms and their impact on traditional software development: A systematic literature review," *Int. J. Software Engineering and Applications*, vol. 15, no. 2, pp. 1–18, 2024.

[4]. E. Martinez and L. Pfister, "Benefits and limitations of using low-code development to support digitalization in the construction industry," *Automation in Construction*, vol. 152, Art. no. 104909, 2023, doi: 10.1016/j.autcon.2023.104909.

[5]. S. A. A. Naqvi and P. Drews, "Understanding low-code evolution, adoption, and ecosystems for software development," in *Proc. 15th Int. Conf. Software Business (ICSOB 2024)*, Utrecht, The Netherlands, Nov. 18–20, 2024, pp. 1–15.

[6]. D. Di Ruscio, D. Kolovos, J. de Lara, A. Pierantonio, M. Tisi, and M. Wimmer, "Low-code development and model-driven engineering: Two sides of the same coin?" *Software and Systems Modeling*, vol. 21, no. 2, pp. 437–446, 2022, doi: 10.1007/s10270-021-00970-2.

[7]. R. C. Domingues Filho, M. J. Silva, and M. L. M. Marinho, "Low-code quality metrics for agile software development," in *Proc. XXIII Brazilian Symp. Software Quality (SBQS 2024)*, Salvador, Brazil, Nov. 5–8, 2024, pp. 1–9, doi: 10.1145/3701625.3701626.

[8]. M. A. Rahman and H. H. Alqahtani, "New perspectives for cyber security in software development when end-user development meets artificial intelligence," *IEEE Access*, vol. 11, pp. 102345–102358, 2023.

[9]. J. Varajão and H. Fernandes, "Measuring productivity in low-code software development," *Information and Software Technology*, vol. 141, Art. no. 106735, 2022, doi: 10.1016/j.infsof.2021.106735.

International Journal of Advanced Research in Computer and Communication Engineering
Impact Factor 8.471 ⁜ Peer-reviewed & Refereed journal ⁜ Vol. 15, Issue 3, March 2026
DOI: 10.17148/IJARCCE.2026.153145

[10]. V. Verma and S. Shree, "Low-code and no-code platforms: From concept to creation," *Int. J. Engineering Research & Technology (IJERT)*, vol. 14, no. 1, pp. 1–8, Jan. 2025.
[11]. R. Picek, "Low-code/no-code platforms and modern ERP systems," in *Proc. 2023 Int. Conf. Information Management (ICIM)*, Oxford, U.K., 2023, pp. 44–49, doi: 10.1109/ICIM58774.2023.00014.
[12]. N. Rao, J. Tsay, K. Kate, V. J. Hellendoorn, and M. Hirzel, "AI for low-code for AI," in *Proc. 29th Int. Conf. Intelligent User Interfaces (IUI '24)*, Greenville, SC, USA, Mar. 18–21, 2024, pp. 1–16, doi: 10.1145/3640543.3645203.
[13]. H. E. Okeke and O. D. Akinbolajo, "Integrating AI and automation into low-code development: Opportunities and challenges," *Int. J. Science and Research Archive*, vol. 8, no. 1, pp. 1094–1109, 2023, doi: 10.30574/ijsra.2023.8.1.0077.
[14]. Gartner, *2026 CIO Agenda Preview: Succeeding When Plans Change—Again*, Gartner, Stamford, CT, USA, 2025.
[15]. D. Begonha, A. Kopper, and T. Thirakul, *Low-code/no-code: A way to transform shadow IT into a next-generation technology asset*, McKinsey & Company, New York, NY, USA, Aug. 2022.
[16]. V. Viswanadhapalli, "The future of intelligent automation: How low-code/no-code platforms are transforming AI decisioning," *Int. J. Engineering and Computer Science*, vol. 14, no. 1, pp. 26803–26872, 2025.
[17]. Z. Xu *et al.*, "ComfyUI-R1: Exploring reasoning models for workflow generation," arXiv:2506.09790v1, Jun. 2025. [Online]. Available: https://arxiv.org/abs/2506.09790. Accessed: Feb. 1, 2026.
[18]. PwC, *2024 Cloud and AI Business Survey: Building the AI-Powered Business*, PricewaterhouseCoopers, London, U.K., 2024.
[19]. Gartner, *Low-code and no-code development technologies: Market trends and adoption insights*, Gartner, Stamford, CT, USA, 2024.
[20]. Gartner, *From Generative AI to Agentic AI: Delivering Measurable ROI*, Gartner Research Note, Stamford, CT, USA, 2025.
[21]. G. Pace, "Pace-layered application strategy," Gartner, Stamford, CT, USA. [Online]. Available: https://www.gartner.com/en/documents. Accessed: Feb. 1, 2026.
[22]. A. Cockburn, "Hexagonal architecture," *Alistair Cockburn Blog*, 2005. [Online]. Available: https://alistair.cockburn.us/hexagonal-architecture. Accessed: Feb. 1, 2026.

